UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

PROGRAMA DE MAESTRÍA Y DOCTORADO EN INGENIERÍA
MAESTRÍA EN INGENIERÍA MECÁNICA –MECATRÓNICA

PROTOTIPO ASRS

TESIS
QUE PARA OPTAR POR EL GRADO DE:
MAESTRO EN INGENIERÍA

PRESENTA:
ING. LEONARDO DANIEL CASTILLO ARISTA

TUTOR PRINCIPAL
DR. ALEJANDRO C. RAMÍREZ REIVICH, FACULTAD DE INGENIERÍA , UNAM

CIUDAD UNIVERSITARIA, CD. MX., NOVIEMBRE 2023

JURADO ASIGNADO

Presidente:	Dr. González Villela Víctor Javier
Secretario:	Dr. Borja Ramírez Vicente
1 er. Vocal:	Dr. Ramírez Reivich Alejandro C.
2 do. Vocal:	Dra. Corona Lira María Del Pilar
3 er. Vocal:	Dr. Ávila Cedillo Javier Noé

Lugar o lugares donde se realizó la tesis: Ciudad de México

TUTOR DE TESIS:

Dr. Ramírez Reivich Alejandro C.

FIRMA

DEDICATORIA

A mi madre, Fabiola Arista por siempre motivarme a continuar mis estudios, a aspirar a tener una mejor vida, valores y ser un hombre de bien.

A mi padre, Martín Castillo Cervantes, por ser un excelente ejemplo de hombre honrado, honorable y trabajador, por apoyarme en mi decisión de continuar mis estudios y por todo el soporte que he recibido.

A mi familia, por siempre motivarme a mejorar, porque sé que cuento con ustedes ¡¡en especial a usted abuelita q. e. p. d. … Gracias!!

A mi pareja, Tania, por el soporte, la motivación, el apoyo que me has dado, en fin, por todo el amor que he recibido y por ser parte de cada fase de este trabajo, si alguien sabe por lo que pase fuiste tú, ¡¡gracias por todo!!

Gracias a todos por nunca dejarme solo en mis problemas, nunca lo olvidaré.

AGRADECIMIENTOS

Al CONACYT por el apoyo económico durante mi estancia en el posgrado.

A la UNAM y la Facultad de Ingeniería por su labor, la pase muy bien en las clases y lo disfruté.

Al Dr. González Villela Víctor Javier porque gracias a su clase de diseño mecatrónico descubrí los sistemas embebidos y con ello lo que se ha vuelto mi fuente de trabajo y quizá mi hobby favorito, ¡gracias!

A mi tutor el Dr. Alejando Ramírez Reivich por permitirme libertad creativa en mi tesis, y motivar a los alumnos a innovar y crear maquinas, ¡gracias!

Se agradece al apoyo Número de PAPIIT - IT102122, Diseño y construcción de un dispositivo y banco de pruebas para evaluar el daño efectuado a patrimonio cultural e histórico de México durante su desmontaje, preparación, traslado, montaje y exhibición. Donde se usaron instrumentos para validar el funcionamiento del sistema.

RESUMEN

Cada año se producen nuevos materiales hemerográficos y bibliográficos impresos, los cuales requieren de espacio de almacenaje adicional, la biblioteca y hemeroteca nacionales de México son las encargadas de resguardar y poner a disposición pública la totalidad de la producción editorial mexicana, la cual constituye el patrimonio bibliográfico y hemerográfico de la nación, actualmente el espacio disponible para continuar almacenando estos materiales se encuentra agotados, es por eso que se requiere implementar alguna solución para continuar el almacenaje.

Los sistemas ASRS son sistemas automatizados de almacenamiento y recuperación que maximizan el espacio de almacenaje.

El presente trabajo expone la creación de un prototipo de estos sistemas que permita evaluar las variables involucradas en el sistema, asi como identificar las áreas críticas de diseño, proponer la ingeniería del sistema inicial y una primera versión de software embebido para el control del prototipo.

Se expone el diseño, pruebas del prototipo y reflexiones sobre esta primera iteración para continuar con la mejora en áreas como el software, diseño mecánico y electrónico.

Contenido

JURADO ASIGNADO ii
DEDICATORIA iii
AGRADECIMIENTOS iii
RESUMEN iv
ÍNDICE DE TABLAS vii
ÍNDICE DE IMÁGENES viii
1 INTRODUCCIÓN: 1
2 ESTRUCTURA DE ESTE DOCUMENTO 1
3 OBJETIVOS Y ALCANCE: 2
3.1 Objetivo 2
3.2 Alcance 2
4 ANTECEDENTES: 3
5 METODOLOGIA/DESCRIPCIÓN DEL PROCESO 6
6 JUSTIFICACIÓN 7
7 DISEÑO 8
7.1 INTERFAZ DEL BIBLIOTECARIO: 9
7.2 PROTOTIPO FÍSICO 12
7.3 COMUNICACIONES 13
7.4 RACKS Y ESTRUCTURAS 13
7.5 Descripción De Las Necesidades 14
8 Elementos Constitutivos De Un Sistema Mecatrónico 15
8.1 Módulo de ensamble y actuación (Diseño Mecánico) 17
8.1.1 Robot 17
8.1.2 Extractor 17
8.1.3 Tren inferior 20
8.1.4 Columna 21
8.2 Módulo de Medición 23
8.3 Módulo de Medio Ambiente 26
8.4 Módulo de Procesador 27
8.5 Módulo de Software 29
8.5.1 Módulo de Software del robot: 29

8.5.2 Módulo de Software del microcontrolador del módulo de comunicación 29
8.5.3 Módulo de Software de la interfaz de control del usuario: 30
8.5.4 Paradigma de desarrollo implementada.....30
8.5.5 Patrón de Proxy de Hardware:.....30
8.5.6 Patrón Observador: 32
8.6 Módulo de Interfaces 34
8.7 Módulo de Comunicación.....35
8.8 Paradigma de Funcionamiento.....46
8.9 Inicialización del Sistema:.....47
8.10 Petición de Usuario: 47
8.10.1 Petición de Recuperación: 47
8.10.2 Petición de Resguardo: 47
8.10.3 Petición de Reubicación:.....48
8.10.4 Modificar Parámetros:.....48
8.10.5 Secuencia de Cancelación.....48
8.10.6 Gestión de Unidades y Base de datos.....48
8.11 Redundancia.....49
9 CONSTRUCCIÓN O IMPLEMENTACIÓN DEL PROTOTIPO 51
10 REALIZACIÓN DE PRUEBAS Y ADQUISICIÓN DE DATOS 58
10.1 Test Cumplimiento de la Secuencia de Recuperación: 58
10.1.1 Test Cumplimiento de la Secuencia de Resguardo 63
10.2 Test Cumplimiento de la Secuencia de Reubicación.....67
11 DOCUMENTACIÓN Y REPORTE DE RESULTADOS 72
11.1 Software: 72
11.2 Hardware:.....73
11.3 Diseño Mecánico 74
11.4 Ingeniería de Sistemas.....76
12 CONCLUSIONES Y COMENTARIOS 78
13 TRABAJO FUTURO 81
14 ANEXOS 85
14.1 Anexo A: Detalle de la implementacion del patrón de suscripción en lenguaje C. 85
15 Referencias.....88

ÍNDICE DE TABLAS

Tabla 1: Componentes del extractor _____ *19*
Tabla 2: Componentes tren inferior _____ *20*
Tabla 3: Componentes del ensamble de columna _____ *22*
*Tabla 4 : Especificaciones del sensor seleccionado [16]*_____ *24*
Tabla 5: Comandos para comunicarse con el sensor [16] _____ *24*
Tabla 6: Codificación de la Información a utilizar _____ *38*
Tabla 7: Codificación de la Información _____ *40*
Tabla 8: Codificación de la Información _____ *41*
Tabla 9: Codificación de la Información _____ *44*
*Tabla 10: Coordenadas seleccionadas para el test (recuperación) en cada eje de movimiento*_____ *58*
*Tabla 11: Test de inicialización*_____ *59*
*Tabla 12: Test recepción de peticiones*_____ *60*
Tabla 13: Test primera secuencia de movimiento completa _____ *60*
Tabla 14: Test reporte estado después de la primera fase de movimiento _____ *61*
Tabla 15: Test segunda fase de movimiento _____ *61*
*Tabla 16: Test reporte estado después de la segunda fase de movimiento*_____ *62*
*Tabla 17: Test finalización de operaciones*_____ *62*
Tabla 18: Coordenadas seleccionadas para el test (resguardo) en cada eje de movimiento _____ *63*
*Tabla 19: Test de inicialización*_____ *63*
*Tabla 20: Test recepción de peticiones*_____ *64*
Tabla 21: Test primera secuencia de movimiento completa _____ *65*
Tabla 22: Test reporte estado después de la primera fase de movimiento _____ *65*
Tabla 23: Test segunda fase de movimiento _____ *66*
*Tabla 24: Test reporte estado después de la segunda fase de movimiento*_____ *66*
*Tabla 25: Test finalización de operaciones*_____ *67*
Tabla 26: Coordenadas seleccionadas para el test (reubicación) en cada eje de movimiento _____ *67*
*Tabla 27: Test de inicialización*_____ *68*
*Tabla 28: Test recepción de peticiones*_____ *68*
Tabla 29: Test primera secuencia de movimiento completa _____ *69*
Tabla 30: Test reporte estado después de la primera fase de movimiento _____ *70*
Tabla 31: Test segunda fase de movimiento _____ *70*
*Tabla 32: Test reporte estado después de la segunda fase de movimiento*_____ *71*
*Tabla 33: Test finalización de operaciones*_____ *71*

ÍNDICE DE IMÁGENES

Figura 1:Transelevador de columna [19] _____ *4*
Figura 2: Robot Móvil [20] _____ *4*
Figura 3: Diagrama de los 4 bloques fundamentales que componen el prototipo _____ *9*
Figura 4: Diagrama de los elementos constitutivos de un sistema mecatrónico [6]. ____ *15*
Figura 5: Vista lateral del extractor de unidades _____ *18*
Figura 6: Vista frontal del extractor de unidades _____ *18*
Figura 7: Tren inferior _____ *20*
Figura 8: Ensamble de columna _____ *21*
Figura 9: Principio de funcionamiento de un sensor TOF [16] _____ *23*
Figura 10. TF Mini Plus [16] _____ *25*
Figura 11: Diagrama del Sensor óptico de final de carrera seleccionado _____ *25*
Figura 12: Diagrama de comunicaciones STM32F466RE _____ *27*
Figura 13: Diagrama de comunicaciones STM32F103C8 _____ *27*
Figura 14: Diagrama UML del patrón de proxy de hardware _____ *32*
Figura 15: Diagrama UML del patrón observador _____ *33*
Figura 16: Diagrama de los protocolos de comunicación utilizados _____ *35*
Figura 17: Diagrama del flujo de información saliente de los microcontroladores del prototipo _____ *36*
Figura 18: Diagrama de Comunicación 1 _____ *39*
Figura 19: Diagrama de comunicación 2 _____ *40*
Figura 20: Diagrama de Comunicación 3 _____ *42*
Figura 21: Diagrama de Comunicación 4 _____ *45*
Figura 22: Diagrama del Paradigma de funcionamiento general. _____ *46*
Figura 23: Materiales de Construcción _____ *51*
Figura 24: Tarjeta que tuvo que ser rediseñada _____ *52*
Figura 25: Diagrama Electrónico del Convertidor de voltaje TTL usando MOSFETS _____ *52*
Figura 26: Vista de algunas Piezas producidas en impresión 3d _____ *53*
Figura 27: Instalación del módulo de control del eje Y _____ *54*
Figura 28: Vista del gabinete siendo cableado _____ *55*
Figura 29: Vista del gabinete finalizado _____ *56*
Figura 30: Prototipo finalizado _____ *57*
Figura 31: Módulo Z en pruebas de funcionamiento. _____ *58*
Figura 32: Mecanismo telescópico totalmente extendido _____ *76*

1 INTRODUCCIÓN:

La biblioteca y hemeroteca nacionales de México son las encargadas de resguardar y poner a disposición pública la totalidad de la producción editorial mexicana, la cual constituye el patrimonio bibliográfico y hemerográfico de la nación.

Actualmente la biblioteca y hemeroteca nacionales de México, cuyas instalaciones se encuentran dentro de Ciudad Universitaria en la Ciudad de México, sufren un problema de saturación; sus espacios se encuentran al 85% de capacidad.

A partir de esta necesidad es que se plantea la construcción del Centro de Preservación Documental (CPD), el cual se llevará a cabo en el campus Juriquilla de la UNAM ubicado en el estado de Querétaro.

El CPD es un inmueble destinado al resguardo de las unidades documentales **(Edificio de Almacenamiento)** en condiciones óptimas que promuevan su conservación y acceso a largo plazo. Esta edificación deberá cubrir las necesidades de almacenamiento actuales y el crecimiento del acervo proyectado al menos a 25 años a partir de la fecha de ocupación. Se deberá contemplar un crecimiento de este inmueble de manera modular con una proyección a 100 años.

Para aprovechar al máximo el espacio disponible y teniendo en cuenta el crecimiento futuro del acervo y la continua necesidad de espacio de almacenaje, es que se propone automatizar en su totalidad el CPD.

Para llevar a cabo esta tarea se desarrollará un sistema automatizado de almacenamiento y recuperación o AS/RS por sus siglas en inglés

2 ESTRUCTURA DE ESTE DOCUMENTO

Capítulo 3: Describe los objetivos y alcance de este documento.

Capítulo 4: Provee una descripción de los antecedentes y el marco teórico sobre el que este documento se desarrolla.

Capítulo 5: Provee una descripción de las metodologías usadas para desarrollar el prototipo

Capítulo 6: Provee la descripción de la necesidad de la realización del prototipo y de la importancia de llevarlo a cabo.

Capítulo 7: Establece el marco del diseño del prototipo, en sus pilares fundamentales: software, mecánica, electrónica y comunicaciones.

Capítulo 8: Se desarrolla el proceso de diseño para cada elemento constitutivo del sistema mecatrónico (el prototipo ASRS en sí mismo).

Capítulo 9: Provee una descripción de la fase de construcción del prototipo.

Capítulo 10: Presenta los resultados de las pruebas hechas al prototipo.

Capítulo 11: Provee un análisis de los resultados de las pruebas y observaciones hechas durante el funcionamiento del prototipo.

Capítulo 12: Presenta las conclusiones y observaciones finales del trabajo desarrollado.

Capítulo 13: Aquí se exponen las áreas de mejora identificadas para continuar desarrollando el producto.

Capítulo 14: Presenta los anexos e información útil, aplicada durante el desarrollo del presente trabajo.

3 OBJETIVOS Y ALCANCE:

3.1 Objetivo

El objetivo de esta tesis es desarrollar un prototipo de sistema automatizado, cuyo fin es , entender los principales aspectos inherentes a este, como son: el intercambio de información entre subsistemas, la secuencia de movimientos necesarios, entender el flujo de entradas y salidas y la manera correcta de gestionarlas para que los movimientos del robot sean seguros y repetibles; servir como banco de pruebas para analizar la incorporación de diferentes componentes, subsistemas y estudiar la mejor manera de integrar los servicios y protocolos de intercambio de información. Cubrir las principales características que debe reunir un prototipo como son:

- Ser un enfoque rápido e iterativo para el desarrollo del sistema.
- Poder ser modificable y reutilizable con requerimientos añadidos.
- Ser una valiosa herramienta para verificar la viabilidad del sistema y analizar si los requisitos analizados son suficientes para llevar a cabo el sistema principal.
- Bajo costo de construcción.
- Demostrar la integración de los distintos componentes o subsistemas del producto
- Simular rápida y flexiblemente el comportamiento o cambio en las propiedades del producto o alguna de sus partes, mediante la modificación de ciertos parámetros o variables.
- Reducir significativamente el riego de generar iteraciones costosas.
- Obtener datos que permiten analizar, generar y completar decisiones.

3.2 Alcance

El alcance contemplado es la generación de un sistema de prototipo base que permita estudiar las variables involucradas de mejor manera en la construcción del sistema real, así como identificar las necesidades primordiales y métodos de solución, así como la generación de las primeras versiones de especificaciones técnicas en los pilares

fundamentales que requiere uno de estos sistemas para funcionar como lo es el diseño mecánico, el software y el diseño eléctrico electrónico, por lo que el alcance contemplado incluye:

- Generación de requerimientos y especificaciones.
- Construcción de un prototipo funcional.
- Creación de una HMI para interactuar con el prototipo.
- Pruebas funcionales.

4 ANTECEDENTES:

Los sistemas AS/RS son ampliamente utilizados en diferentes industrias alrededor del mundo, permiten que las industrias y demás aplicaciones, donde son implementados, mejoren el rendimiento, la capacidad de almacenaje y reduzcan los costos operativos, ya que permiten mover mayores volúmenes de producto y aprovechar al máximo el espacio de almacenamiento [1].

Existen numerosas soluciones comerciales de sistemas ASRS, todas tienen el mismo propósito que es maximizar la utilización del espacio de almacenaje, pero pueden ser diferenciados en cuanto al método de extracción o recuperación y entrega de mercancías, capacidad de carga y principios operativos.

A continuación, se presentan algunos de los conceptos más representativos y utilizados en la industria:

- ***Transelevadores de columna.***

Este concepto necesita de un riel instalado en el piso, para trasladarse (Puede ser uno o dos), es el que posee la mayor capacidad de carga y el dispositivo de extracción con el que cuentan puede cambiar el principio de funcionamiento, según la aplicación que se requiera y puede ser arrastre, carga, neumático, etc. Como su nombre lo indica, constructivamente estos transelevadores pueden poseer una o dos columnas, por medio de las cuales se traslada el peso de la carga al tren inferior y se desplaza el mecanismo de extracción a la altura deseada. Son bastante robustos y pueden alcanzar alturas considerables.

A la derecha se muestra la fotografía de un transelevador de columna comercial.

Figura 1:Transelevador de columna [19]

- **Robótica Móvil**

La robótica móvil posee una mayor versatilidad que los transelevadores, pues conjuntan los elementos que compone un sistema ASRS en una sola unidad de pequeño tamaño, aunque las prestaciones que poseen estos sistemas suelen ser significativamente menores que la de los transelevadores, pues al ser un sistema integrado, sacrifican atributos como la velocidad de desplazamiento, la capacidad de carga y la altura de desplazamiento. Para hacer eficiente la velocidad de entrega muchas veces se requiere más de un solo robot y pueden requerir instalar accesorios extra en los racks para poder desplazarse sobre ellos que añaden costos de implementación. En cuanto a las ventajas, no requieren un sistema de Conveyor o robots auxiliares para llevar el paquete solicitado al usuario final, pues al ser un robot móvil es capaz de desempeñar esta tarea, pueden operar simultáneamente para aumentar la eficiencia el proceso de recuperación de mercancías, integran en una unidad compacta los tres elementos modulares de un transelevador convencional, los cuales son: El mecanismo de recuperación o extractor, el elemento de soporte estructural o columna y el tren inferior encargado de mover el transelevador. A la derecha se muestra un ejemplo de esta clase de robots.

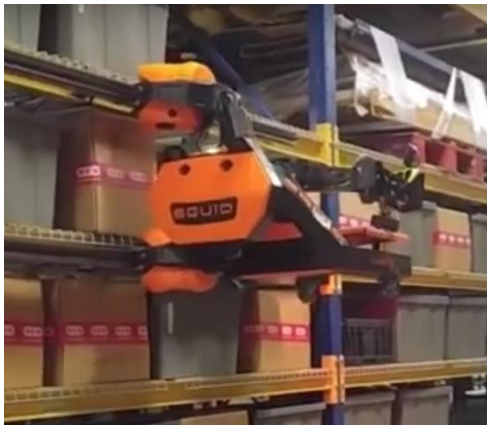


Figura 2: Robot Móvil [20]

Combinaciones de estos dos conceptos se han realizado en la academia, por ejemplo, en [2], fabricaron un prototipo cuyo propósito es extraer libros de un estante, los cuales se encuentran guardados dentro de una carcasa protectora de metal, el cual tenía como mecanismo de extracción un robot manipulador, podía moverse como un robot móvil y además posee una columna, la cual eleva el robot manipulador.

En India tuvo lugar otro desarrollo [3], cuyo fin principal era demostrar el diseño de un sistema robótico inteligente para una biblioteca, donde la identificación del material se hacía a través de RFID y la totalidad del prototipo se basó en el popular entorno de desarrollo ARDUINO®. El prototipo contempló una base de datos y registro de usuarios, así como un brazo robótico para la extracción de libros y un robot seguidor de línea como tren motriz del prototipo.

Otro prototipo desarrollado en India [4] con características similares al prototipo expuesto anteriormente, también ocupa la identificación de materiales mediante tecnología RFID, pero diferente método de elevación del mecanismo extractor, pues en este prototipo se utiliza el método de trabajo virtual, utilizado en los elevadores de tijera y al igual que el anterior, utiliza un brazo robótico para extraer los materiales.

En [5] se plantea un sistema de soporte para toma de decisiones, el cual puede ser utilizado para diseñar, gestionar y controlar los sistemas de un almacén. Con este sistema se implementa una metodología que considere el diseño estratégico del almacén y la gestión de operaciones. También permite simular el desempeño logístico y el desempeño en cuanto al manejo del material dentro del almacén. La herramienta permite almacenar información y elaborar un conjunto de datos orientados a la solución de diseño y configuración del almacén.

En [6] se habla de lo importante que son los sistemas AS/RS en almacenes modernos, ya que ofrecen ventajas como un mejor control del inventario, optimización del tiempo de trabajo en el almacén, espacio y equipo. Muchos de los problemas y enfoques relacionados a la eficiencia de AS/RS son considerados en este trabajo. Este artículo presenta una visión general de la literatura relacionado con el AS/RS en los últimos 40 años. Presenta una descripción del actual estado del arte y discute prospectos del futuro. El propósito de este capítulo es proveer a los investigadores y diseñadores de entendimiento sobre cómo aplicar los enfoques existentes de manera eficiente.

En [7] se diseña una infraestructura para la gestión de información y mecanismos clave para gestionar una biblioteca. También se discute la innovación respecto a la adquisición de información, la gestión rápida de los libros y su entrega. La tecnología RFID fue el núcleo en este trabajo. Se propone toda una arquitectura basada en RFID, el cual será establecido como un SAAS (Software como Servicio), una plataforma computacional y un conjunto de sistema integrado de gestión de información de la biblioteca basado en Internet de las Cosas

(IoT). El documento propone un servicio de gestión basado en Internet, especialmente para el manejo de claves, aplicaciones que proporcionan servicios de logística y aplicaciones de control.

En [8] se propone un nuevo modelo para gestionar una biblioteca mediante un sistema robótico. El robot es capaz de recolectar los libros solicitados por el usuario y mediante el algoritmo desarrollado, en la interfaz gráfica, el robot sugiere material recomendado/relacionado a las preferencias del usuario. Toda la transferencia de información se da mediante tecnología Wi-Fi para minimizar el tiempo de búsqueda del material solicitado y el tiempo que tarden los bibliotecarios en encontrar los libros.

En [9] se presenta una etapa de prueba completa basada en LabVIEW para la asociación de bibliotecas. El robot fue formado, fabricado y certificado. Se utilizaron artefactos embebidos para estudiantes para los sistemas de operación y control. También se utilizó tecnología RFID para distinción de todo el material bibliográfico.

En [10] se habla del uso de identificación por radio frecuencia (RFID por sus siglas en inglés) aplicado en el acomodo de libros dentro de una biblioteca. En esta investigación se obtuvo una precisión razonable respecto al control de la posición de los libros, saber exactamente dónde se ubicaban en tiempo real. Mencionan que el área de mejoría se presenta en los lectores RFID, desarrollar lectores más robustos y que puedan escanear la información al mismo tiempo que el robot está en movimiento.

5 METODOLOGIA/DESCRIPCIÓN DEL PROCESO

La metodología base del desarrollo del prototipo es la sugerida en “Innovación de producto” [11] específicamente el apartado titulado “Realización de prototipos”, aunque se han tomado puntos muy específicos de otras metodologías como son “Metodología en el desarrollo de productos” [12] y “Systems Engineering” [13] [14].

La metodología base seleccionada establece una serie de puntos que deben llevarse a cabo para la realización de prototipos, los cuales se reproducen a continuación:

1.- Identificar la razón por la que se requiere desarrollar un prototipo: elementos de las especificaciones del producto a verificar, parámetros se desea manipular; datos que se requieren obtener y rango en que debe funcionar el prototipo, etc.

2.- Definir el tipo de prototipo a desarrollar: funcional, matemático, físico, simulación con software, etc.

3.- Establecer la manera en que se pueden manipular y medir las variables a estudiar

4.- Diseñar el prototipo que se requiere.

5.- Construcción o implementación del prototipo.

6.- Realización de pruebas y adquisición de datos.

7.- Documentar y reportar los resultados de las pruebas.

8.- Retroalimentar al proceso de innovación de producto.

6 JUSTIFICACIÓN

Para llevar a cabo la automatización del CPD se requerirá diseñar e implementar un complejo sistema automatizado de almacenamiento y recuperación, que operará de manera autónoma con apenas la intervención humana necesaria, por lo que es importante entender completamente el comportamiento del sistema y las variables involucradas, así como la interacción entre subsistemas durante la operación del mismo.

Debido a la necesidad expresada anteriormente y con el fin de lograr un entendimiento profundo de todos los parámetros involucrados para hacer que el robot opere de manera segura y sin contratiempos se ha propuesto desarrollar un “prototipo físico de desarrollo”, el cual se utilizara para probar conceptos, realizar pruebas de comportamiento, identificar aspectos para optimización y determinar el correcto flujo de información que tiene que tener lugar desde que se realiza una petición al robot, hasta que este entrega la unidad solicitada.

Los principales parámetros a manipular y obtener son las entradas y salidas del sistema, ¿y qué son estas entradas y salidas a la que nos referimos? Por entradas nos referimos a señales provenientes de los Sensores de los diferentes subsistemas que comprenden el prototipo a desarrollar, señales que tienen una gran importancia y deben ser interpretadas en un lapso de tiempo concreto para asegurar un comportamiento preciso y repetible del robot. Estas señales también tienen que ser interpretadas correctamente en orden de aparición, duración y amplitud, lo cual llevará al comportamiento deseado y seguro del sistema completo; también como entrada tenemos la información recibida por parte de la persona que hace una petición al robot, la cual debe ser interpretada para ejecutar la respuesta correspondiente. ¿Y quién se encarga de ejecutar dichas respuestas? La respuesta es las salidas, las cuales hacen referencia a los actuadores que componen el prototipo, por ejemplo, motores; sin embargo, también por salida nos referimos nuevamente a señales de sensores, que en este contexto son un indicador que una tarea ha sido llevada a cabo con éxito o no se ha cumplido, lo que podría llevar a desarrollar un sistema de depuración que permita encontrar errores debido a un mal funcionamiento de hardware que pudiese ser implementado en el sistema real.

Como se mencionó anteriormente la denominación del prototipo es “prototipo físico de desarrollo”, el cual se puede descomponer en sus tres principales pilares constructivos que son:

- La parte mecánica: Que no es otra cosa que la estructura, componentes de rodadura, chumaceras, sujetadores, motores y rieles, encargados de suministrar potencia y guiar los movimientos del prototipo.

- La parte eléctrica/electrónica: A esta parte también podríamos denominarla Hardware, encargada de suministrar la potencia eléctrica y brindar los medios físicos necesarios para el control y las comunicaciones a instalar en el prototipo.

- Las instrucciones de control/Software: Encargado de gestionar las señales enviadas por el hardware para tomar decisiones, administrar la información y ordenar al hardware y mecanismos los movimientos necesarios para cumplir una orden recibida.

7 DISEÑO

Debido a que el propósito del prototipo es servir como una primera aproximación que permita estudiar las variables involucradas en un complejo sistema ASRS, se proponen a continuación una serie de funciones que como primera iteración se ha pensado cumplen las necesidades que cubren esta clase de sistemas y se han añadido algunas características que hacen que sea un producto deseable y que desempeñe un comportamiento repetible y seguro.

El desarrollar cada función llevará a identificar las especificaciones que tiene que tener el prototipo [15].

Dichas funciones pueden ser resumidas en 4 bloques fundamentales:

Interfaz del Bibliotecario, Comunicaciones, Prototipo Físico, Racks y Estructuras.

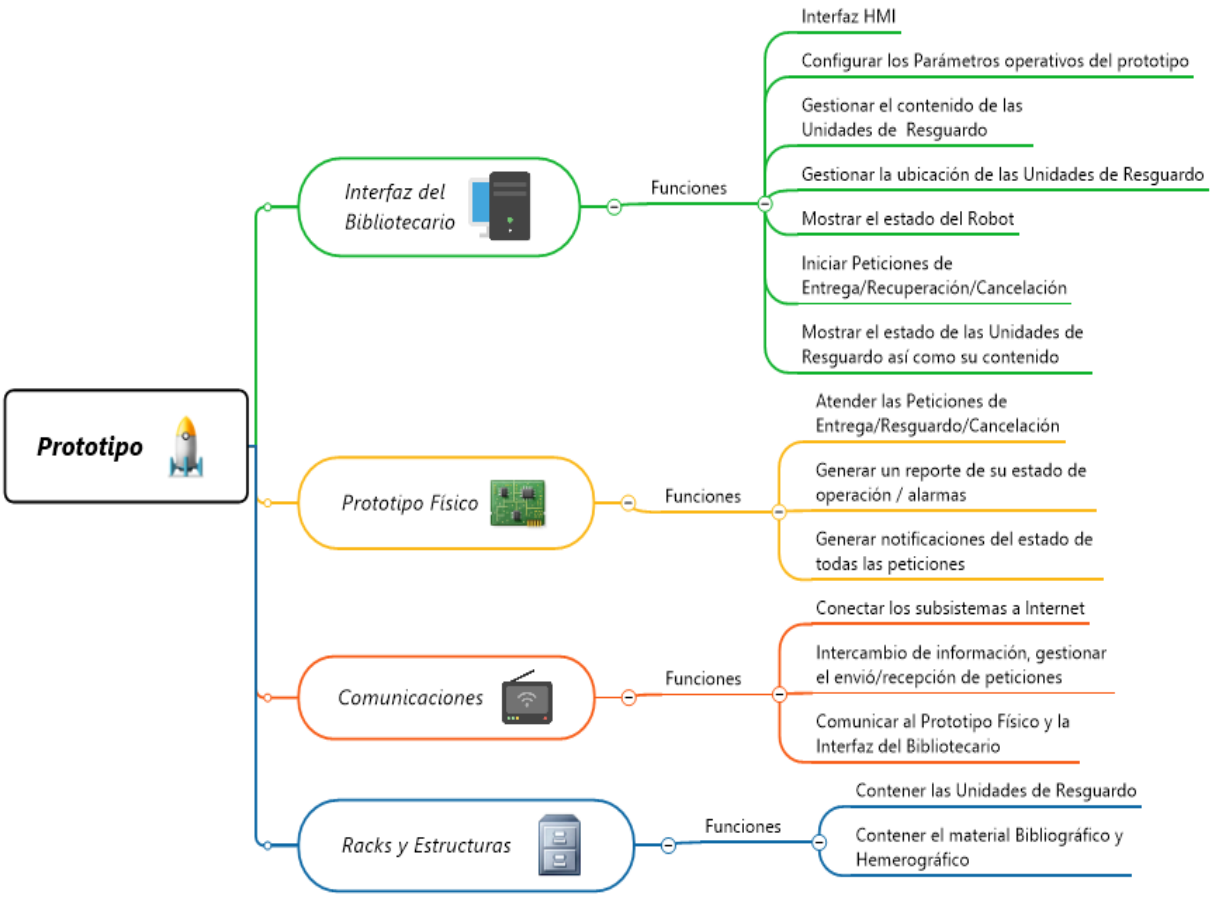


Figura 3: Diagrama de los 4 bloques fundamentales que componen el prototipo

A continuación, se explican las funciones que tiene que cumplir cada bloque funcional y se profundiza en algunas definiciones necesarias:

7.1 INTERFAZ DEL BIBLIOTECARIO:

***Interfaz HMI**:* La interfaz HMI es el principal y más importante medio de interacción entre el operador y el prototipo, representa una manera amigable para el usuario de gestionar las tareas requeridas y el intercambio de información necesario para empezar una tarea o darla por concluida.

***Configuración de Parámetros Operativos**:* Esta configuración se realiza mediante la interfaz HMI, el usuario configura los parámetros operativos principales del robot y los demás sistemas de entrega, como son:

-Velocidad: Aumento o disminución de la velocidad dentro de márgenes permitidos de los distintos desplazamientos posibles del robot y sistemas de entrega en los tres ejes: *X, Y, Z.*

-Dimensiones: Configura las dimensiones de los espacios vacíos en los racks, variable que es utilizada por el robot para controlar el movimiento. Se requiere de tres dimensiones que son: Altura (Y), Anchura (X) y Profundidad (Z).

Gestión del Contenido de las Unidades de Resguardo*:* Por medio de esta función es que el usuario puede gestionar el contenido de las unidades de resguardo, se derivan de ella tres funciones principales:

-Agregar: Con esta función se agregan títulos, los cuales estarán contenidos dentro de la unidad de resguardo. Fundamental en el acomodo inicial de unidades de resguardo.

-Eliminar: Elimina títulos contenidos dentro de la unidad.

-Cambio: Con esta función se puede realizar el cambio de títulos de una unidad a otra unidad según se requiera.

Gestión de la Ubicación de las Unidades de Resguardo: Con esta función se gestiona la ubicación de las unidades de resguardo en los racks, esta función genera un código hexadecimal que es enviado al robot cuando se inicia una solicitud de extracción de unidad de los racks que sirve para identificar sus coordenadas en el rack. Se conforma de tres funciones principales:

-Agregar: Con esta función se agregan unidades de almacenaje, las cuales estarán contenidas dentro de los Racks. Fundamental en el acomodo inicial de unidades de resguardo.

-Eliminar: Elimina unidades existentes dentro de un Rack.

-Cambio: Con esta función se puede realizar el cambio de posición de unidades dentro del Rack.

Mostrar el Estado del Robot*:* Esta función está pensada para que el usuario se entere del estado del Robot, muestre en pantalla variables como la velocidad de operación de los subsistemas, el estado (si está libre para atender una nueva

petición, o si está ejecutando alguna petición y de ser así que clase de petición se encuentra ejecutando). Se puede decir que esta es una función de solo lectura o informativa por lo que no es posible interactuar con el robot a través de ella.

***Iniciar Peticiones de Entrega/ Recuperación/ Cancelación**:* Todas las Peticiones generadas por el usuario serán gestionadas a través de esta función cuyo propósito principal es indicarle al robot y los sistemas de entrega que hacer con las unidades de resguardo, a continuación, se describen cada una de estas Peticiones.

Petición de Entrega: Cuando el usuario requiere de algún título contenido en alguna unidad de resguardo, se genera esta petición.
La petición es ignorada si el robot tiene activa alguna alarma, el paro de emergencia se encuentra activo o si se encuentra procesando alguna otra solicitud, por lo que el robot esperara hasta que su estado se encuentre como libre y a la espera de una petición.

Petición de Resguardo: Cuando el usuario requiere almacenar nuevamente algún título dentro de alguna unidad de resguardo, se genera esta petición.
La petición es ignorada si el robot tiene activa alguna alarma, el paro de emergencia se encuentra activo o si se encuentra procesando alguna otra solicitud, por lo que el robot esperara hasta que su estado se encuentre como libre y a la espera de una petición.

Petición de Cancelación: Cuando el usuario requiere cancelar alguna petición ordenada con anterioridad se ejecuta esta petición.
Si alguna petición es cancelada la unidad de resguardo se devuelve al estado anterior en el cual se encontraba.

Mostrar el Estado de las Unidades de Resguardo y su Contenido: Esta función permite monitorear el estado de las unidades de resguardo, así como de su contenido, es decir, permite saber si una unidad de resguardo se encuentra fuera del espacio asignado para ella en el rack, si se encuentra en el rack, o si se encuentra a disposición del usuario final. Lo mismo sucede con el contenido de la unidad, informa si algún título contenido originalmente dentro de la unidad ha sido prestado, se encuentra dentro de su unidad o si está en mantenimiento, etc.

Esta función está estrechamente relacionada con otras funciones mencionadas anteriormente, ya que esta función permite obtener el estado de una unidad de resguardo, así como de su contenido, por lo tanto, tiene la capacidad de bloquear la ejecución de otras funciones, por ejemplo, no se puede ejecutar una petición de

entrega si el estado de la unidad se encuentra “entregado”, ni tampoco se puede solicitar una petición de recuperación de una unidad que ya se encuentra en el rack.

Respecto al contenido es bastante similar lo que pasa, se informa que el ejemplar ya ha sido extraído de la unidad para su respectivo préstamo, consulta o mantenimiento, de ese modo se evitaría una petición de entrega innecesaria.

7.2 PROTOTIPO FÍSICO

***Atender Peticiones de Entrega/Resguardo/Cancelación**:* Estas funciones se ejecutan respectivamente según la petición recibida de la Interfaz del Bibliotecario, una vez que son recibidas y si el robot no se encuentra ejecutando ya alguna otra petición se ejecutan las siguientes secuencias:

Secuencia de Entrega

Esta función comienza con la petición de un usuario de un libro o archivo resguardado en un contenedor o unidad logística.

La petición es enviada al prototipo a través de internet, el prototipo recibe la petición y atiende a ella, moviéndose a las coordenadas donde se encuentra la unidad de resguardo (X, Y, Z) dentro del rack, una vez en posición el robot procede a accionar el mecanismo de extracción de unidades de resguardo con el que está equipado, toma la unidad y la lleva al punto de entrega, una vez que la entrega se completó, el prototipo va a su posición HOME y espera una nueva petición, ya sea de entrega o de resguardo de una unidad entregada previamente.

Secuencia de Recuperación

Esta función de recuperación, tiene lugar cuando es necesario almacenar nuevamente la unidad de resguardo que ya no es requerida, aunque hay otro momento importante en el cual la secuencia de recuperación tendrá lugar y ese es el acomodo inicial de las unidades de almacenaje en los racks correspondientes.

Inicia cuando, desde la interfaz del Bibliotecario, se envía una solicitud de recuperación al robot, los sistemas de entrega la llevan entonces hasta los racks donde el robot la acomoda en el lugar correspondiente al número de unidad enviado junto con la solicitud de recuperación.

Secuencia de Cancelación

La secuencia de cancelación se lleva a cabo cuando por alguna razón no se requiere completar la solicitud o hubo un error en dicha solicitud, el robot

cancela la operación y regresa la unidad al rack o nuevamente al sistema de entrega para que vuelva al Bibliotecario.

Generar reporte del Estado de Operación/Alarmas activas: Con esta función el prototipo genera un reporte de su estado de operación, indica la posición en la que se encuentra, la velocidad de los actuadores, sensores activos y las banderas de operación requeridas para iniciar la atención a las Peticiones provenientes de la Interfaz del Bibliotecario como son: "Libre" , que indica que el robot puede atender peticiones; "Ocupado", que indica que el robot se encuentra atendiendo ya alguna petición y "Cancelación en Curso", que indica que se ha interrumpido una petición y el robot está atendiendo la petición de cancelación.

También se genera el reporte de alarmas, el cual tiene la capacidad de bloquear parcial o totalmente la operación del robot, dichas alarmas se detallan más adelante.

Generar Notificaciones del Estado de las Peticiones: Con esta función se monitorea el estado de una petición, se muestra si está en proceso o si ha sido concluida o cancelada.

7.3 COMUNICACIONES

Conectar los Subsistemas a Internet: Es necesario que los subsistemas que componen el prototipo tengan conexión a internet, de ese modo se crea un respaldo de las operaciones realizadas y se comunican los subsistemas.

Intercambio de Información, Gestionar el Envío/Recepción de Peticiones: Con esta función se asegura que la información haya sido correctamente enviada y recibida, por el correspondiente emisor y destinatario.

Comunicar al Prototipo Físico y la Interfaz del Bibliotecario: Esta función está estrechamente relacionada con las dos anteriores, establece el principal canal de comunicación entre estos dos subsistemas del prototipo y contiene parámetros para saber cuándo la comunicación ha sido exitosa o hay un fallo en la información enviada o recibida.

7.4 RACKS Y ESTRUCTURAS

Contener el Material Bibliográfico y Hemerográfico: Esta función corresponde a las Unidades de Resguardo, las cuales contienen y resguardan todo el material de

consulta gestionado por la Biblioteca. Su función es proteger a los materiales durante el traslado y durante su estancia en los Racks.

Contener las Unidades de Resguardo: Esta función corresponde a los Racks, los cuales contienen y resguardan las Unidades de Resguardo mientras no se requiere alguno de sus títulos para consulta o mantenimiento.

7.5 Descripción De Las Necesidades

En este punto se han mencionado las principales funciones que tiene que cumplir cada uno de los subsistemas que componen el prototipo. Esto sirve como un excelente punto de partida para identificar los métodos de solución que pueden ser aplicables con el fin de que todas las funciones requeridas por el prototipo puedan ser ejecutadas de la mejor manera posible.

También se han identificado las soluciones que tienen que implementarse para que la relación entre las entradas y salidas de cada función, que debe cumplir cada subsistema, se lleven a cabo.

Ya que conseguimos llegar a este punto en momento de implementar la metodología de desarrollo de productos mecatrónicos [6].

8 Elementos Constitutivos De Un Sistema Mecatrónico

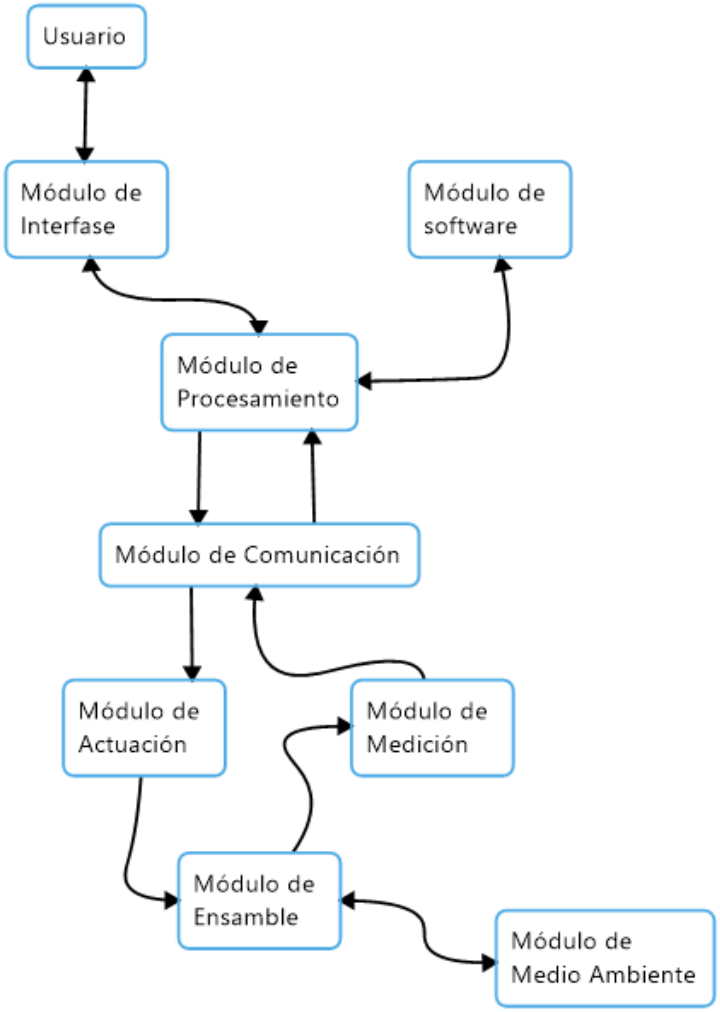


Figura 4: Diagrama de los elementos constitutivos de un sistema mecatrónico [6].

Módulo de medio ambiente: Se refiere a todos los parámetros externos que afectaran las operaciones del sistema mecatrónico, tales como la temperatura, carga del sistema, humedad, etc. Constituye los parámetros frontera dentro de los cuales el sistema debe funcionar.

Módulo de ensamble: Representa los elementos mecánicos y estructurales del sistema. Se relaciona con el comportamiento estructural y las propiedades de los materiales del sistema, así como el contexto. Las entradas a este módulo son los movimientos proporcionados por el módulo de actuación. El sistema de ensamble

además se relaciona directamente con la apariencia del sistema, es por eso que debe ser considerada la estética del sistema dentro de este módulo.

Módulo de Actuación: Representa la fuerza de acción del sistema. Mediante este módulo se cambian las condiciones del sistema. Las condiciones de entrada están establecidas por la salida del módulo del procesadore y las salidas se definen por el tipo de movimiento requerido.

Módulo de medición: Se relaciona con la obtención de información sobre el comportamiento del sistema. Como parámetros de entrada se tienen propiedades físicas del módulo de ensamble, mientras que como salidas se tiene la información a ser transmitida.

Módulo de Comunicación: Encargado de transmitir la información entre módulos dentro del sistema. Las condiciones de entrada y salida se relacionan con la naturaleza de la información que se requiere transmitir, la distancia a la cual será transmitida y el ambiente de operación.

Módulo del Procesador: Relacionado con el procesamiento de la información proveniente del módulo de medición y el de interfaces. Como parámetros de entrada se tienen los parámetros medidos y el establecimiento de demandas junto a los parámetros del sistema, tales como la velocidad de operación. La salida determina la operación del módulo de actuación y proporciona información al módulo de interfaces.

Módulo de Software: Contiene las instrucciones de operación y algoritmos definidos para el sistema y el control de las operaciones del módulo de procesamiento. La naturaleza y forma de este módulo están ligadas al correspondiente módulo de procesamiento.

Módulo de Interfaces: Esta relacionado con la transferencia de información entre los diferentes subsistemas del sistema mecatrónico, proporciona la interfaz hombre-máquina para el intercambio de información entre el sistema y el usuario. Las entradas y salidas están relacionadas con la naturaleza de la transferencia de información involucrada.
En el proceso de diseño la naturaleza de las funciones a ser proporcionadas por cada módulo individual y para cada nivel pueden ser establecidas, así como la naturaleza de la información transmitida pero no necesariamente la forma de la información.

En este momento es posible hacer analogías entre los módulos que componen un sistema mecatrónico expuestos anteriormente y los subsistemas que componen el prototipo asi como la interacción que tienen entre ellos y las funciones que requieren cumplir.

8.1 Módulo de ensamble y actuación (Diseño Mecánico)

8.1.1 Robot

El diseño mecánico del robot se ha inspirado en los principios operativos de modelos comerciales de esta clase de sistemas y se ha pensado para que sea capaz de cumplir con las funciones descritas anteriormente.

El diseño mecánico del prototipo puede ser fragmentado en tres partes primordiales, las cuales se describen a continuación: Extractor, Columna, Tren inferior.

8.1.2 Extractor

Función: Es el encargado de extraer la unidad solicitada por el usuario del rack, en el cual se encuentra alojada y de volver a introducirla cuando el usuario ha terminado el uso del material correspondiente.

Existen diversas configuraciones posibles, el punto de partida seleccionado para implementar en el prototipo es un extractor de arrastre, el extractor cuenta con una transmisión de cadena, las cual tienen unos aditamentos de tipo pin que hacen posible arrastrar la unidad al extractor por medio de unas cejillas dispuestas en la unidad de resguardo, los pines entran al interior de estas cejillas y tiran de esta, guiándola a la plataforma dispuesta en el extractor para soportarla. Este mecanismo permite maximizar el espacio de almacenaje, ya que no requiere espacio adicional al que ya de por si ocupa la unidad de almacenaje en el rack para realizar operaciones.

El extractor se pretende sea modular, así con este mismo prototipo podría probarse otra clase de métodos para realizar el trabajo como por ejemplo los extractores telescópicos.

El trabajo de arrastre es desarrollado por dos motores, los cuales giran en sentido contrario para realizar una entrega e invierten respectivamente sus direcciones de giro cuando tienen que cumplir una petición de recuperación.

El extractor dispone además de un mecanismo de piñón y cremallera, el cual es responsable de acercar el extractor a los racks para poder arrastrar las unidades logísticas. Este mecanismo de piñón y cremallera cuenta con su propio motor.

Como un complemento se ha añadido un sistema de contrapeso para disminuir la carga requerida por el motor de la columna, responsable de los movimientos de ascenso y descenso del mecanismo telescópico.

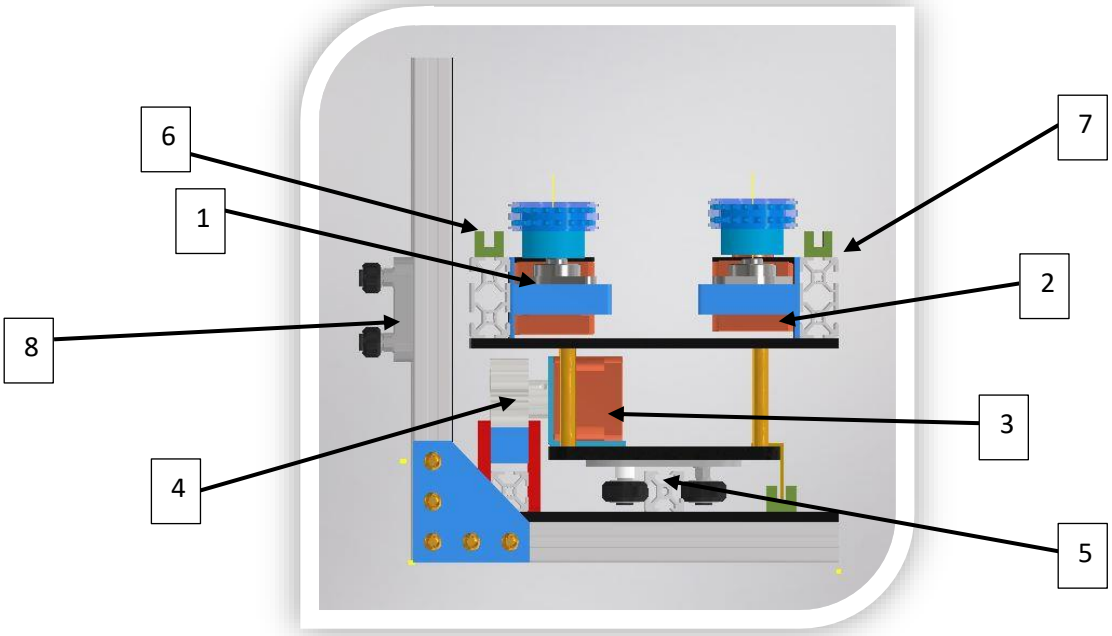


Figura 5: Vista lateral del extractor de unidades

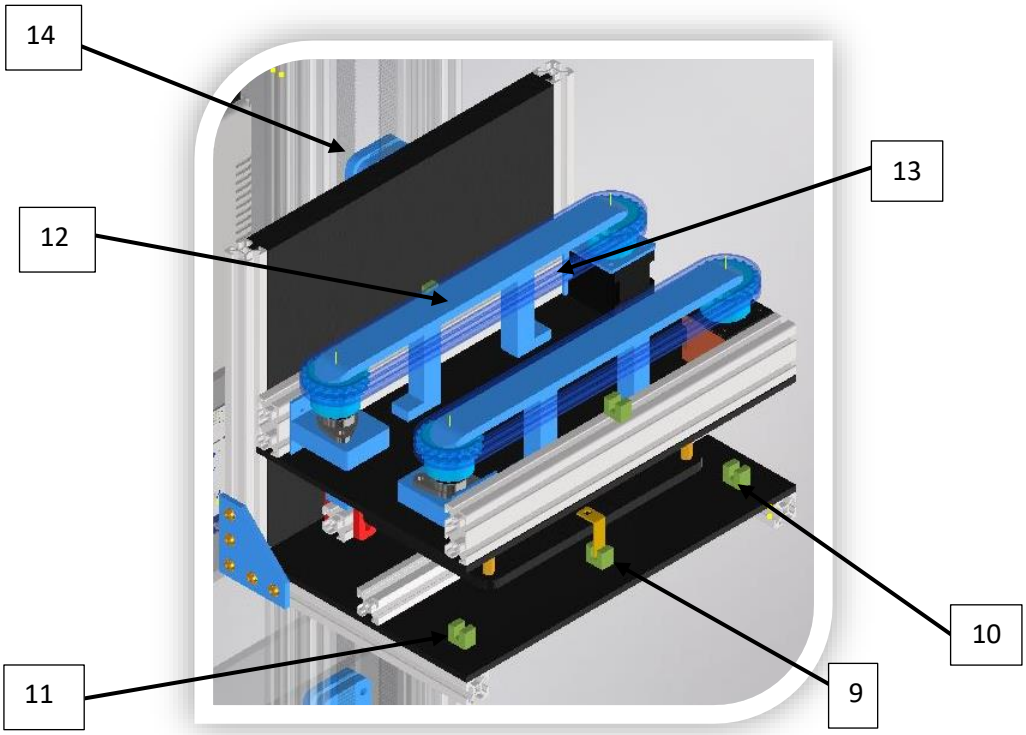


Figura 6: Vista frontal del extractor de unidades

Núm.	Nombre	Función
1	***Motor Extractor 1***	Proveer la potencia necesaria para arrastrar la caja hacia el interior del mecanismo extractor o expulsarla del mismo según corresponda.
2	***Motor Extractor 2***	Proveer la potencia necesaria para arrastrar la caja hacia el interior del mecanismo extractor o expulsarla del mismo según corresponda.
3	***Motor del Mecanismo telescópico***	Proveer la potencia necesaria para mover extender y retraer el mecanismo hacia y desde los racks donde se encuentras las unidades de resguardo.
4	***Mecanismo de piñón-cremallera***	Realizar el movimiento telescópico requerido.
5	***Barra guía y plataforma deslizante***	Guiar el movimiento telescópico
6	***Sensor 1 home del pin del mecanismo extractor***	Sensar la posición del Pin 1
7	***Sensor 2 home del pin del mecanismo extractor***	Sensar la posición del Pin 2
8	***Plataforma deslizante***	Guiar el movimiento ascendente y descendente del extractor de unidades.
9	***Sensor home del mecanismo telescópico***	Sensar la posición inicial del mecanismo telescópico
10	***Sensor de final de carrera del mecanismo telescópico 1***	Detener el funcionamiento del motor del mecanismo telescópico cuando este ha alcanzado su destino.
11	***Sensor de final de carrera del mecanismo telescópico 2***	Detener el funcionamiento del motor del mecanismo telescópico cuando este ha alcanzado su destino.
12	***Plataforma de soporte para las unidades de resguardo***	Soportar las unidades de almacenaje en su traslado.
13	***Cadena de arrastre***	A estas cadenas se ha adjuntado un pin de arrastre, el cual se mueve con la cadena y su función es llevar la unidad al interior del extractor o fuera de este según sea el caso.
14	***Placas de sujeción***	Sujetar al extractor a la polea dentada encargada de realizar el movimiento ascendente y descendente del mismo.

Tabla 1: Componentes del extractor

8.1.3 Tren inferior

Función: Desplazar el prototipo longitudinalmente sobre el eje X. El tren inferior debe tener la capacidad de resistir la carga total del robot, asi como proveer los mecanismos necesarios para que este pueda desplazarse a lo largo de un pasillo. Consta de un solo motor principal, cuya transmisión es un arreglo de poleas para maximizar el torque producido por el motor y vencer la fuerza de fricción producida por el total de la carga del robot. Consta de unas ruedas guía, las cuales en contacto con guías laterales mantienen la trayectoria rectilínea de todo el conjunto cuando este se desplaza longitudinalmente.

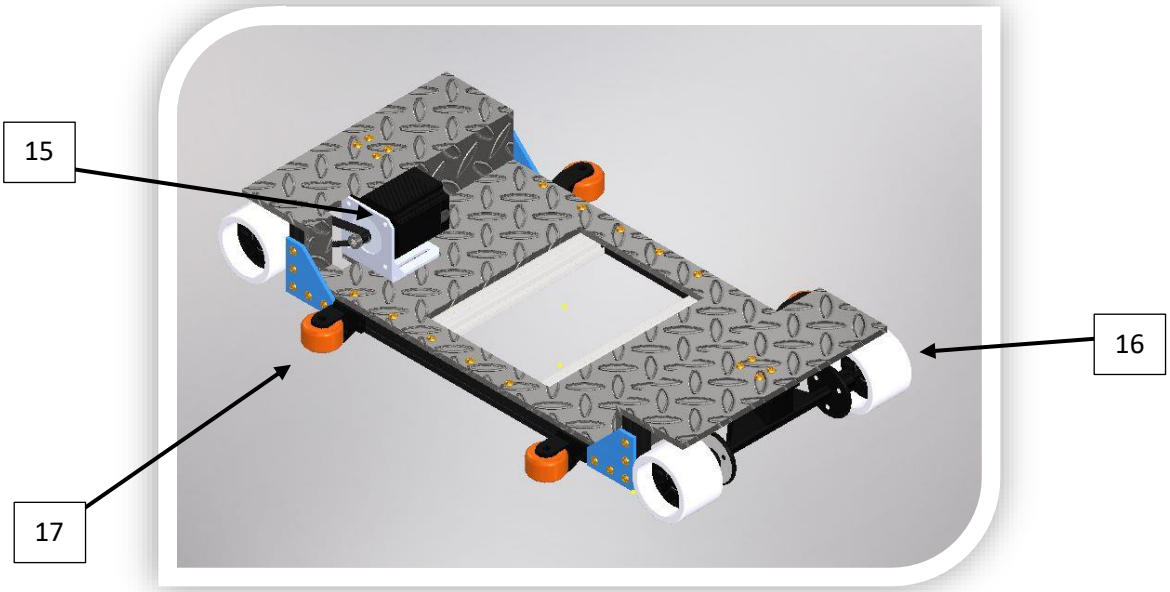


Figura 7: Tren inferior

Núm.	Nombre	Función
15	***Motor Eje X***	Proveer la potencia necesaria para desplazar el prototipo a lo largo del pasillo. NEMA 23
16	***Tren Motriz***	Proveer el soporte principal del robot, asi como los mecanismos de movimiento requeridos para realizar un desplazamiento longitudinal.
17	***Ruedas Guía***	Proveer la potencia necesaria para mover extender y retraer el mecanismo hacia y desde los racks donde se encuentras las unidades de resguardo.

Tabla 2: Componentes tren inferior

8.1.4 Columna

Función: Es la encargada de guiar el movimiento ascendente y descendente del extractor, así como del respectivo contrapeso necesario para reducir la carga teórica que tendría que mover el motor destinado para tal propósito, también sirve como medio para que transmitir el movimiento de traslación del tren inferior al extractor. El movimiento ascendente y descendente del extractor se lleva a cabo por una trasmisión de cadena instalada en la columna.

Otra función importante de la columna es albergar los componentes de control electrónicos, en el centro de ella se ha instalado el tablero de control del robot.

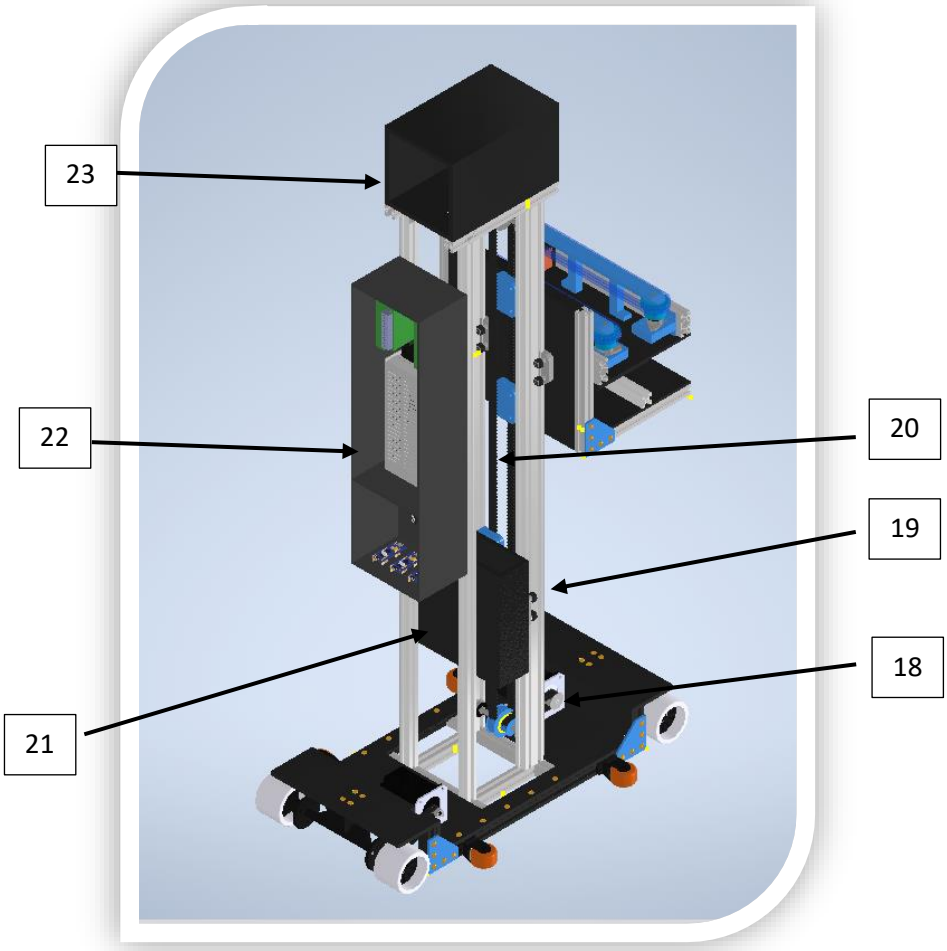


Figura 8: Ensamble de columna

Núm.	Nombre	Función
18	***Motor Eje Y***	Proveer la potencia necesaria para desplazar el prototipo a lo largo del pasillo. NEMA 23
19	***Columnas***	Proveer el soporte del extractor, asi como servir de guía para el desplazamiento vertical del mismo "Eje Y", también brinda el soporte para el contrapeso y al igual que con el extractor sirve de guía para el anteriormente mencionado. También sirve como base para la instalación del gabinete de control, donde se albergan la mayoría de los componentes electrónicos que hacen funcional el prototipo.
20	***Polea dentada***	Convierte el movimiento de rotación del motor del eje Y, en movimiento ascendente-descendente para el extractor y el contrapeso. Tanto el contrapeso como el extractor se encuentran unidos a la polea para moverse junto con ella en direcciones opuestas en cada ciclo.
21	***Contrapeso***	Sirve como ventaja mecánica. Utilizando el contrapeso, es que el motor ahora solo tiene que vencer la fuerza de fricción existente entre la columna que sirve como elemento guía y las plataformas deslizantes con las que cuenta tanto el extractor como el contrapeso.
22	***Gabinete Control***	Almacena el hardware del robot. Aquí es que se han colocado las fuentes, tarjetas de control asi como los drivers de los motores.
23	***Gabinete de Control 2***	Espacio extra de almacenaje.

Tabla 3: Componentes del ensamble de columna

8.2 Módulo de Medición

El módulo de medición es el encargado de sensar la posición del robot en todo momento, mediante este es que se conocen los parámetros operativos y la respectiva etapa que está siendo desarrollada de una determinada tarea.

Posición: Como sabemos el robot tiene movimiento en los tres ejes de simetría, la posición se calculará de la siguiente manera para cada uno de ellos:

Eje X: Se refiere al desplazamiento longitudinal que llevara a cabo el robot (a lo largo del pasillo). Esta posición será medida mediante un sensor de tiempo de vuelo.
También se instalarán dos sensores mecánicos de final de carrera para indicar que la longitud máxima ha sido recorrida.

Eje Y: Se refiere al desplazamiento vertical del extractor (a lo largo de la altura). Esta posición será medida mediante un sensor de tiempo de vuelo.
También se instalarán dos sensores mecánicos de final de carrera para indicar que la longitud máxima ha sido recorrida.

Para sensar la posición del robot en los ejes coordenados "X" e "Y", se ha seleccionado un sensor LiDAR que monitorea la posición del robot en los ejes antes mencionados. Los sensores LiDAR han sido clave en el desarrollo de la robótica los últimos años, ampliamente utilizados en drones, robots manipuladores, sistemas como el que se pretende desarrollar ASRS, la industria automotriz como los son lo son los sistemas avanzados de asistencias a la conducción, ADAS por sus siglas en inglés, etc.

El principio de funcionamiento es el siguiente:

Los sensores de tiempo de vuelo [16], TOF por sus siglas en inglés, emiten ondas de modulación de rayos infrarrojos cercanos de forma periódica, que se reflejarán al sensor después de entrar en contacto con algún objeto. El sensor calcula el tiempo de vuelo de la señal midiendo la diferencia de fase de ida y vuelta y luego calcula el rango relativo entre el producto y el objeto de detección, como se muestra en la Figura:

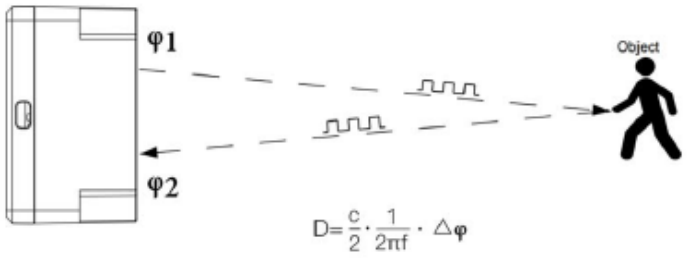


Figura 9: Principio de funcionamiento de un sensor TOF [16]

Se ha seleccionado el sensor TFmini plus de la compañía Benewake, debido a que es una solución de bajo costo que implementa algoritmos de compensación, elevada taza de transferencia de información y además se comunica mediante interfaz UART, además de que varias características del mismo se pueden configurar, como la taza de transferencia de información, si el intercambio de información se realizara de forma automática o a menos que el microcontrolador se lo indique, además de poseer un excelente rango de funcionamiento óptimo para la aplicación (0.1 a 12m) y buena resolución (5mm).

Description	Parameter value
Operating range	0.1m~12m①
Accuracy	±5cm@ (0.1-6m) ②
	±1%@ (6m-12m)
Measurement unit	cm
Range resolution	5mm
FOV	3.6°③
Frame rate	1~1000Hz (adjustable) ④

Tabla 4 : Especificaciones del sensor seleccionado [16]

Parameters	Command	Response	Remark	Default setting
Obtain firmware version	5A 04 01 5F	5A 07 01 V1 V2 V3 SU	Version V3.V2.V1	
System reset	5A 04 02 60	5A 05 02 00 60	Succeeded	
		5A 05 02 01 61	Failed	
Frame rate	5A 06 03 LL HH SU	5A 06 03 LL HH SU	1-1000Hz①	100Hz
Trigger detection	5A 04 04 62	Data frame	After setting the frame rate to 0 ,detection can be triggered with this command	
Output format	5A 05 05 01 65	5A 05 05 01 65	Standard 9 bytes(cm)	√
	5A 05 05 02 66	5A 05 05 02 66	Pixhawk②	/
	5A 05 05 06 6A	5A 05 05 06 6A	Standard 9 bytes(mm)	/
Baud rate	5A 08 06 H1 H2 H3 H4 SU	5A 08 06 H1 H2 H3 H4 SU	Set baud rate®	115200
Enable/Disable output	5A 05 07 00 66	5A 05 07 00 66	Disable data output	/
	5A 05 07 01 67	5A 05 07 01 67	Enable data output	√
Restore factory settings	5A 04 10 6E	5A 05 10 00 6E	Succeeded	
		5A 05 10 01 6F	Failed	
Save settings®	5A 04 11 6F	5A 05 11 00 6F	Succeeded	
		5A 05 11 01 70	Failed	

Tabla 5: Comandos para comunicarse con el sensor [16]

Figura 10. TF Mini Plus [16]

Eje Z: Se refiere a la distancia que es capaz de desplazar el mecanismo telescópico del extractor para acercarse a los pasillos, estas distancias serán medidas mediante finales de carrera mecánicos.

Para sensar la posición en el eje Z, se dispondrá de 5 sensores de tipo interruptor óptico. Dos de ellos para sensar el punto de inicio de los pines de extracción el cual también sirve para saber que los pines han terminado su recorrido y los tres restantes para sensar la posición del mecanismo telescópico. Es suficiente con estos sensores ya la distancia de recorrido tanto para los pines de extracción asi como para el mecanismo telescópico son constantes. Los sensores seleccionados admiten un voltaje de alimentación de 2.7 a 5V y traen embebida la electrónica necesaria para su correcto funcionamiento.

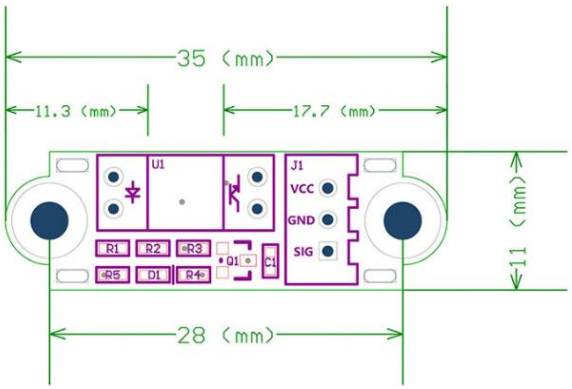


Figura 11: Diagrama del Sensor óptico de final de carrera seleccionado

8.3 Módulo de Medio Ambiente

El módulo de medio ambiente provee severas e importantes restricciones de seguridad para el correcto funcionamiento del sistema. Como se especificó cuando se describió este módulo, constituye los parámetros frontera del medio dentro de los cuales el sistema debe funcionar.

También provee de información acerca de la medición de estos parámetros para monitorear su estado.

Implementación: La placa utilizada para el control del robot trae un sensor de temperatura, el cual se utilizará como principal medio de protección del hardware, que, al exceder el rango permitido, enviará una alerta de lo sucedido.

Notas: El módulo de medio ambiente puede ser extendido con características importantes propias de la aplicación como lo es la medición del nivel de humedad, la temperatura ambiental, la saturación de oxígeno, la presión atmosférica y otras variables importantes para la preservación del material bibliográfico.

También se pueden añadir celdas de carga al robot para calcular la carga a la que está siendo sometido, lo cual notificaría al usuario y al fabricante del uso indebido del robot al sobrepasar los límites de diseño.

Otra implementación importante que puede tenerse dentro del módulo de medio ambiente es la seguridad. Se podría implementar un sistema de detección de personal dentro del área de operación del robot, lo cual bloquearía el funcionamiento de este asi como zonas inseguras o expuestas de energía eléctrica.

8.4 Módulo de Procesador

Para el módulo de procesador se tienen tres elementos encargados de realizar las diferentes tareas de control y procesamiento:

Microcontroladores del Robot. Para el control del movimiento del prototipo se han especificado los siguientes microcontroladores:

STM32F466RE, microcontrolador de 32 bits con elevadas prestaciones y con capacidades de ejecutar RTOS, es el encargado de gestionar la información proveniente del microcontrolador del módulo de comunicación, de cambiar los parámetros de operación provenientes del usuario, y de gestionar las Peticiones de recuperación o resguardo respectivamente comunicándose con los microcontroladores que controlan el movimiento del prototipo en los tres ejes de movimiento X, Y, Z.

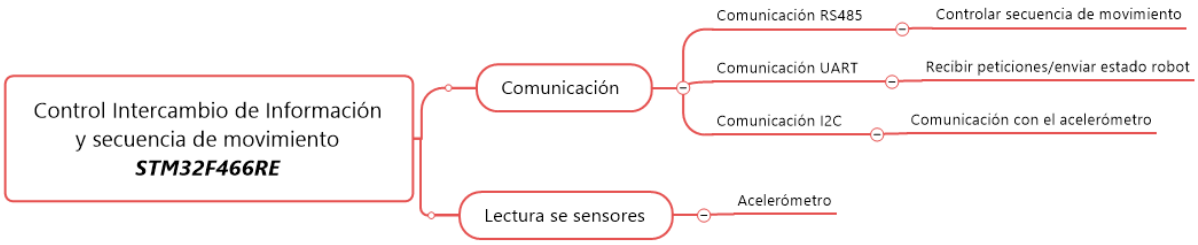


Figura 12: Diagrama de comunicaciones STM32F466RE

STM32F103C8, tres de estos microcontroladores de 32 bits han sido seleccionados para controlar el movimiento en cada uno de los ejes de movimiento. Entre sus funciones se encuentra: *Control de Servomotores, Control de motores a pasos, Lectura de sensores Lidar, lectura de sensores de final de carrera, lectura de acelerómetros, lectura de sensores de temperatura, Finales de carrera ópticos asi como la comunicación con el microcontrolador STM32F466RE.*

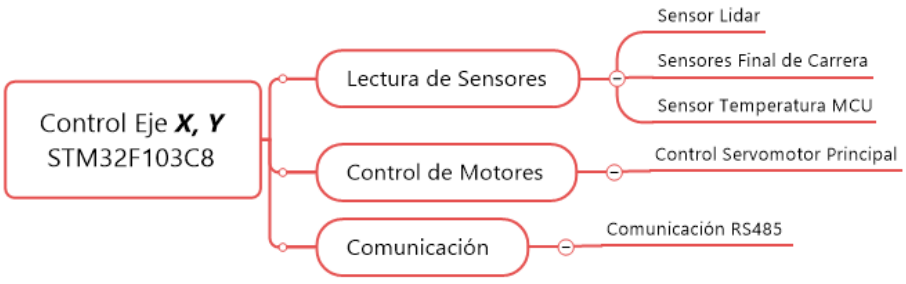


Figura 13: Diagrama de comunicaciones STM32F103C8

Microcontrolador del módulo de comunicación wifi: Para gestionar las comunicaciones se ha utilizado un microcontrolador PIC24 de 16 bits, el cual es el encargado de gestionar la comunicación Wifi del robot, contener el protocolo TCP/IP, así como la gestión de la capa de comunicación con el driver Wifi y el código necesario para implementar el protocolo MQTT, además debe comunicarse con el microcontrolador del robot para intercambiar información entre este y el usuario.

Estación del usuario: es el módulo de procesamiento del prototipo de mayores recursos. Aquí se establecen los parámetros de operación del robot, las peticiones del usuario; se procesa la interfaz gráfica de usuario y se almacenan las ubicaciones de las unidades de resguardo y de su contenido. Tiene comunicación a internet que puede ser mediante Wifi o Ethernet.

8.5 Módulo de Software

Como se puede imaginar, el módulo de Software está sumamente relacionado con el módulo de procesamiento. Cada módulo de procesamiento contiene su propio módulo de software que se encarga de hacer funcionar las respectivas tareas que se mencionan a continuación:

8.5.1 Módulo de Software del robot:

El módulo de software se encarga de procesar los datos recibidos del módulo de medición, así como distribuir esta información a cada microcontrolador encargado de controlar el movimiento en cada eje de movimiento. Controla la coordinación de movimientos del robot además de comunicar el estado de este con el microcontrolador del módulo de comunicación.

Para cumplir con los objetivos previamente mencionados, el módulo de software es específico a cada microcontrolador en el robot.

A continuación, se detallan las principales funciones que tiene que cumplir el módulo de software de cada microcontrolador que controla el movimiento en cada eje del prototipo, así como del microcontrolador que controla la secuencia de movimiento y el intercambio de información entre los diferentes módulos.

Control de movimiento en el eje X (STMF103C8):

8.5.2 Módulo de Software del microcontrolador del módulo de comunicación

Wifi:

Este módulo de software se encarga de gestionar las tareas de comunicación entre el driver Wifi seleccionado para la aplicación y el microcontrolador que a su vez se comunica tanto con la interfaz de usuario como con el robot.

Las funciones que ejecuta este módulo son:

Inicializar Wifi: Establece la conexión al punto de acceso seleccionado, para lo cual se pasa como parámetro el nombre de la red y la contraseña, además cuenta con funciones para la diagnosticar diferentes tipos de errores de comunicación.

Crear un Web Socket: Esta función crea un socket para conectarse a internet y atender los servicios de red requeridos, devuelve el ID del socket creado y al igual que la función anterior tiene mecanismos para detectar errores.

Conectarse al bróker MQTT: Con esta función se establece comunicación entre el microcontrolador y el bróker instalado en el servidor, al cual deberá conectarse también la aplicación HMI para que el sistema quede comunicado y de ese modo

ejecutar todas las prestaciones del protocolo MQTT como suscribirse a tópicos, publicar a tópicos, etc.

Comunicación entre la aplicación HMI con la que interactúa el usuario y el microcontrolador que controla el robot, asi como comunicación del estado de la conexión y las comunicaciones en general.

8.5.3 Módulo de Software de la interfaz de control del usuario:

Este módulo contiene el software de la interfaz HMI, también el control de la base de datos, gestiona la conexión al bróker MQTT, crea la conexión necesaria con el servidor para el almacenaje de datos.

8.5.4 Paradigma de desarrollo implementada

Para implementar el módulo de software se han seleccionado algunos patrones de diseño de software para sistemas embebidos muy populares y de grandes prestaciones los cuales se describen a continuación:

8.5.5 Patrón de Proxy de Hardware:

Con este patrón [17] se encapsula el acceso a todo el hardware del dispositivo independientemente de su implementación física. Es decir, provee código útil para acceder a alguna dependencia de Hardware.

Debe proveer todos los servicios que permitan leer y escribir valores o configuraciones del hardware, así como inicializar o apagar dispositivos según corresponda.

¿Por qué implementar el patrón de proxy de hardware?

Si cada cliente en el prototipo accede a un dispositivo de hardware directamente, los problemas debidos a cambios en el hardware se agravan.

Si cambia la codificación de bits, alguna dirección de memoria o la tecnología de conexión, es necesario modificar a cada cliente dependiente de estos servicios.

Al proporcionar un proxy que sirva como una interfaz entre los clientes y el hardware, el impacto de cambiar algún componente del lado del hardware se ve significativamente disminuido, lo que facilita dichas modificaciones.

La idea de realizar un proyecto que contenga un buen nivel de abstracción de hardware es que pueda ser actualizado fácilmente sin tener que rediseñar toda la aplicación desde el comienzo, así, si se agregan nuevas funcionalidades que demanden un hardware más potente aún se pueda mantener la aplicación con el mínimo de cambios.

El que el proyecto tenga cierta abstracción de hardware implica que la aplicación no tenga que acceder a componentes de determinado hardware, la aplicación "desconoce" el hardware en el que basa su funcionamiento, por eso se dice es abstracto a esta.

Para facilitar el mantenimiento, los clientes del proxy de hardware deben desconocer la codificación, el cifrado y la compresión de bits utilizada por el prototipo. Todos estos detalles deben ser administrados por el proxy de hardware con funciones privadas internas. A continuación, se muestra el diagrama UML (Lenguaje Unificado de Modelado).

El diagrama está constituido por los siguientes elementos:

La interfaz de hardware que es única para cada proxy de hardware, lo cual se indica mediante la asociación bidireccional, lo cual indica que tanto la interfaz de hardware como la clase del proxy de hardware están "conscientes" de la existencia una de la otra y de la relación que tienen entre sí.

A su vez existe una asociación unidireccional entre la clase del cliente de proxy y la clase de proxy de hardware, lo que indica que únicamente el cliente está consciente de la existencia del proxy de hardware e interactúa con él, pero este último desconoce el cliente, el cual es el objetivo deseado.

Dispositivo de Hardware: Este bloque representa el hardware actual. La asociación entre el dispositivo de hardware y el proxy de hardware se realiza mediante una interfaz de software direccionado al hardware mediante direcciones de memoria o interrupciones.

Proxy de Hardware: Esta es la clase primaria en el sistema. Contiene los datos y funciones personalizados para el dispositivo o hardware en cuestión. Por lo general cada dispositivo dispone de su propia función de inicialización, configuración y deshabilitación, las otras funciones proporcionan acceso de lectura a los valores del dispositivo (o hardware en cuestión) o la compartición de datos con el mismo. Aunque en la estructura del patrón se muestra una única función *access (),* pueden existir varias funciones de este tipo, cada una con el nombre del valor que lee o establece.

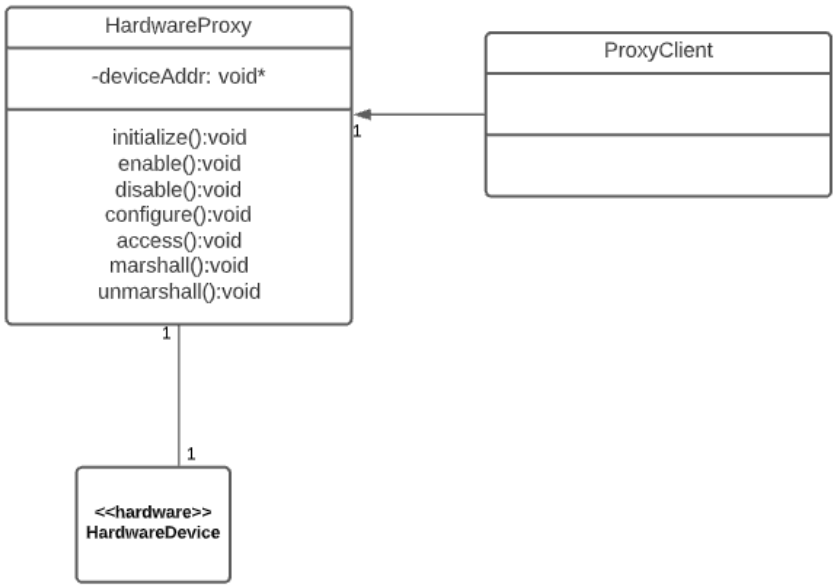


Figura 14: Diagrama UML del patrón de proxy de hardware [17]

8.5.6 Patrón Observador:

El patrón de observador [17] (también conocido como "Patrón de Publicación-Suscripción") proporciona una notificación a un conjunto de clientes interesados en conocer si ciertos datos relevantes provenientes de sensores o módulos han cambiado. Lo hace sin requerir que el servidor que gestiona los datos tenga ningún conocimiento a priori sobre sus clientes. En cambio, los clientes simplemente ejecutan una función de suscripción que les permite agregarse (y eliminarse) dinámicamente a la lista de notificaciones. El servidor puede entonces aplicar cualquier política de notificación que desee. Por lo general, los datos se envían cada vez que llegan datos nuevos, pero los clientes también se pueden actualizar periódicamente.

En una situación en la cual no se ha implementado el Patrón Observador, cada cliente debe solicitar datos periódicamente del servidor que gestiona los datos, en el supuesto de que los datos cambian continuamente (como es el caso en un sistema con sensores de posición, temperatura, cuyos valores están cambiando continuamente) significaría que los clientes tienen que estar constantemente pidiendo actualización de los datos, lo cual es un desperdicio de recursos informáticos y de comunicación, ya que los clientes generalmente no pueden saber cuándo hay nuevos datos disponibles.

El Patrón Observador aborda esta situación, al agregar servicios de suscripción y cancelación de suscripción al servidor de datos. Por lo tanto, un cliente puede

agregarse dinámicamente a la lista de notificaciones sin un conocimiento previo del cliente por parte del servidor. En el lado del servidor, el servidor puede aplicar la política de actualización apropiada para la notificación de sus clientes interesados. Además, el patrón permite la modificación dinámica de las listas de suscriptores y, por lo tanto, agrega una gran flexibilidad al software.

A continuación, se muestra la estructura básica del patrón observador mediante su diagrama UML. *AbstractSubject* es el servidor de datos más la maquinaria necesaria para mantener una lista de suscriptores interesados. Un cliente se puede agregar a la lista de notificaciones al pasar un puntero a una función de aceptación *accept (Datum)* y se elimina llamando a la función de anulación de la suscripción pasando como parámetro el mismo puntero. Cuando *AbstractSubject* decide notificar a sus clientes, la función de notificación *notify ()* recorre la lista de clientes, llamando al puntero a función definido por cada cliente y pasando los datos relevantes como parámetros. *AbstractClient* proporciona la función *accept (Datum)* para recibir y procesar los datos entrantes.

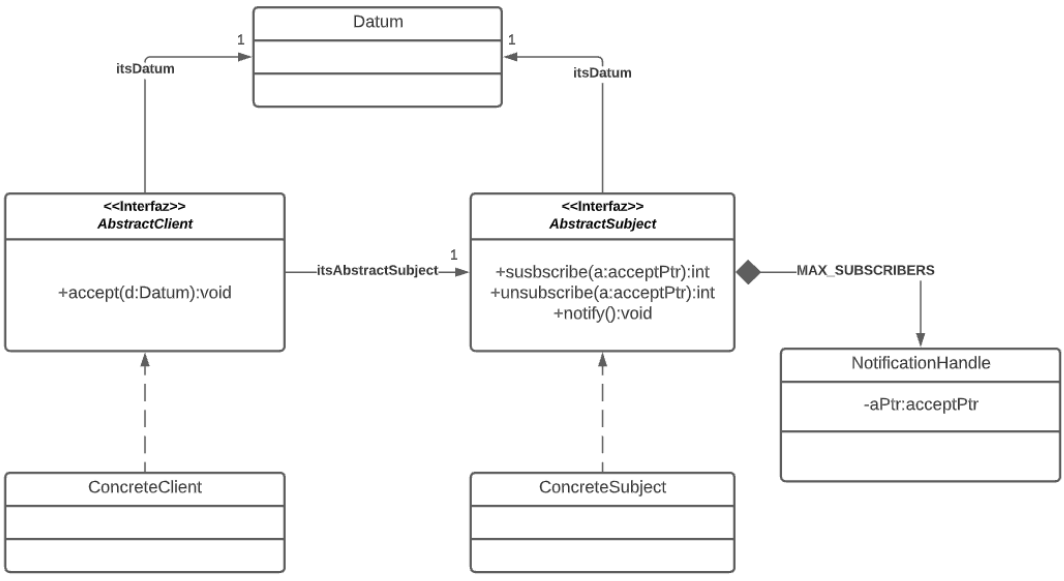


Figura 15: Diagrama UML del patrón observador [17]

Este patrón de diseño fue seleccionado para atender los eventos ocurridos ya que provee una manera clara, ordenada y útil de ordenar el código, atender interrupciones y eventos en una manera centralizada, además de ser muy popular en la industria.

En el prototipo, todo evento interno o externo al microcontrolador es manejado por este patrón, por ejemplo, cuando el sensor Lidar envía al microcontrolador la información de la distancia actual, esto genera una interrupción, la cual es atendida por el servidor que a su vez informa a cada cliente suscrito ejecutando la función registrada por cada cliente para atender este evento, en este caso concreto, existen dos clientes interesados en obtener esta información como lo son: los servomotores, los cuales usan esta información para saber si continuar su movimiento o detenerse, asi mismo esta información es almacenada para su posterior envío a la interfaz gráfica y poder conocer la posición actual en X o Y.

En el anexo A, se detalla la implementación en lenguaje C del Patrón de Suscripción.

8.6 Módulo de Interfaces

El módulo de interfaces es el medio principal de interacción del usuario con el prototipo, a través de este es que el usuario realiza las Peticiones, ve el estado de operación del robot, el estado general de las comunicaciones y el sistema y gestiona el contenido de la base de datos y los parámetros principales para el correcto funcionamiento del sistema.

Diseño del Módulo de interfaz

El módulo de Interfaz debe de contener las siguientes partes:

**Búsqueda de un título: Es importante ya que la búsqueda en la base de datos generalmente será debido a que un usuario requiera un título determinado, el cual está contenido dentro de una unidad de resguardo, al ejecutar la búsqueda debe devolver la unidad en la que se encuentra resguardado.

Almacenar unidad: Esta opción es fundamental en el acomodo inicial de las unidades de resguardo en los racks, se debe ligar el número de unidad con un numero de espacio disponible dentro del rack, entonces se ejecuta una petición de resguardo que almacena dicha unidad en determinado espacio dentro del rack.

Información: Debe mostrar información acerca del estado del módulo de comunicación (intensidad de la señal, sí se encuentra conectado a la red, etc.), estado del robot (a la espera, en una petición); el estado del módulo de medición (lo cual le da al usuario información acerca de la posición del robot) y el estado del módulo de medio ambiente (lo cual brinda información acerca de las características del medio en el cual está operando el robot).

Dentro del módulo de información se tiene la capacidad de introducir el número de una unidad lo cual desplegará información de su contenido y su ubicación en el rack.

Solicitud de petición: En este campo es que el usuario solicita una determinada unidad al robot para su entrega.

Modificar parámetros de operación: Aquí se puede modificar los parámetros operativos del robot, como temperatura máxima del procesador, velocidad de los motores, distancia entre unidades de resguardo, etc.

Solicitar unidad: Esta opción se ejecuta para realizar una petición al robot, una vez que se conoce que en la unidad está el material o título requerido por el usuario.

8.7 Módulo de Comunicación

El módulo de comunicación establece el medio por el cual los diferentes módulos intercambian información, a continuación, se establece el diagrama que muestra el flujo de información en el prototipo:

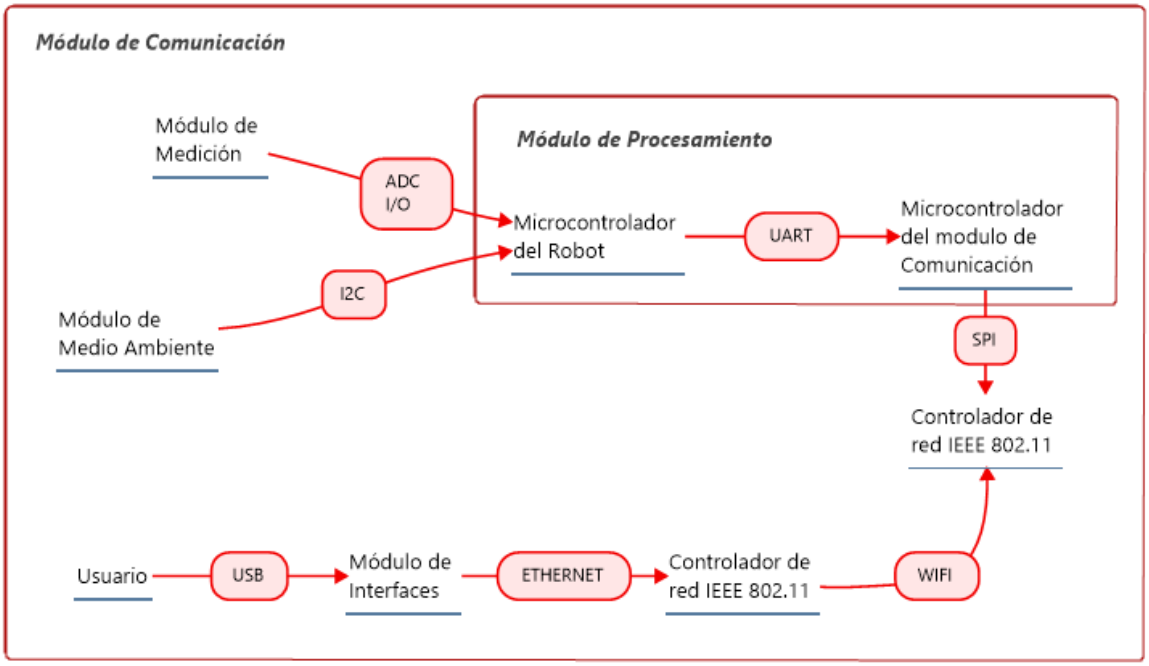


Figura 16: Diagrama de los protocolos de comunicación utilizados

El módulo de comunicación comprende los medios físicos por los cuales se transfiere la información, como se aprecia en el diagrama el flujo de información puede ir del usuario a los diferentes módulos de procesamiento, o de la información recopilada por los módulos de medición y de medio ambiente al usuario.

El flujo de información del Usuario al Robot es el siguiente:

1.- El usuario se comunica con el módulo de interfaces mediante los periféricos del computador como lo son el teclado y el mouse, los cuales se comunican a través de tecnología USB.

2.- La información proveniente del usuario se comunica físicamente mediante ethernet al adaptador de red al que se encuentra conectada la PC utilizada para gestionar el módulo de interfaces a internet.

3.- El adaptador de red que tiene capacidad Wifi, se comunica mediante esta tecnología con el adaptador de red del robot.

4.- Las instrucciones o datos provenientes del usuario y que han sido enviadas a través de wifi, llegan al microcontrolador que gestiona el módulo de comunicación del robot, las cuales se transfieren por medio de UART, al microcontrolador del robot para su respectivo procesamiento.

***El flujo de información del Robot al Usuario es el siguiente**:*

1.- La información generada por los módulos de medición y medio ambiente se transfiere por medio de diferentes periféricos disponibles en el microcontrolador, por ejemplo, algunos sensores utilizan ADC, otros se comunican mediante I2C, mientras que otros solo requieren algunas entradas y salidas estándar (I/O).

2.- El microcontrolador del robot se comunica mediante UART con el microcontrolador del módulo de comunicación y a través de comunicación serial RS485 con los microcontroladores que controlan el movimiento en cada uno de los ejes coordenados.

3.- El microcontrolador del módulo de comunicación transfiere la información mediante SPI al adaptador de red para su respectivo envió por WIFI al adaptador de red al cual se encuentra conectada la PC que contiene el módulo de interfaces.

4.- La información finalmente se transfiere del adaptador de red mediante ethernet a la PC y finalmente dicha información se muestra mediante el módulo de interfaces.

Información Saliente:

A continuación, se muestran los diagramas que muestran gráficamente los datos que envía y recibe el prototipo para el funcionamiento planteado:

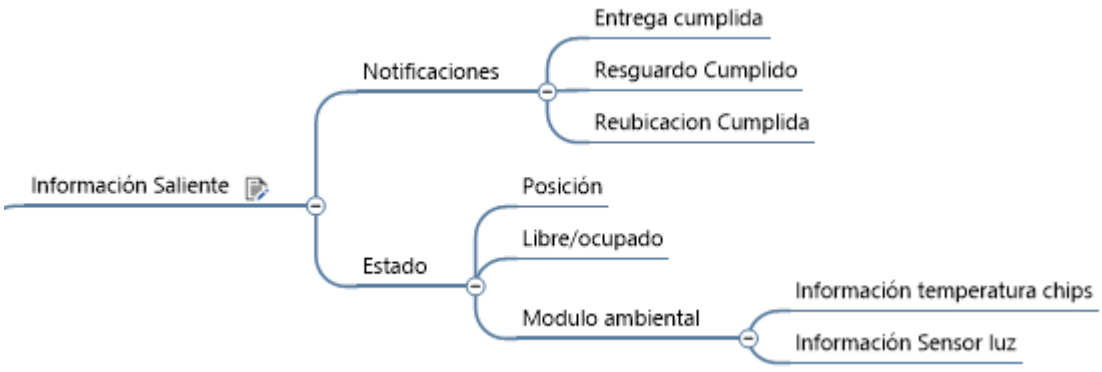


Figura 17: Diagrama del flujo de información saliente de los microcontroladores del prototipo

Notificaciones: El prototipo debe notificar al usuario que se ha cumplido cualquiera de las Peticiones que es capaz de ejecutar, esta información se envía desde el prototipo al módulo de interfaces para ser mostrada al usuario.

Estado: Es importante que el robot pueda comunicar el estado de sí mismo con el usuario, para ello es que el robot debe ser capaz de compartir información como su posición, si es que se encuentra atendiendo alguna petición o está libre, la información de sensores importantes como los sensores de temperatura de los microcontroladores, etc.

El microcontrolador que controla la secuencia de movimiento reúne la información de los otros microcontroladores que controlan el movimiento en los ejes coordenados y la envía al módulo de comunicación para que finalmente sea entregada al usuario, además comunica a cada uno de estos a qué lugar debe moverse, así como las solicitudes de cambio de parámetros operativos (como la velocidad, por ejemplo).

A continuación, se muestra formato de comunicación propuesto, así como el diagrama del mismo para la información que es enviada desde el microcontrolador que controla el movimiento hacia los microcontroladores que mueven el robot:

Byte	Nombre		Valor	Significado
01	Indicar inicio de comunicación		11	NA
02	Indicar con que módulo se quiere comunicar		11	Módulo X
			22	Módulo Y
			33	Módulo Z
03	Indicar la cantidad de bytes a transmitir		8	NA
04	Indicar el tipo de solicitud		65	Modificar Parámetros
			66	Nueva Petición
			67	Solicitud Información (Revisar tabla 2)
05	Primer parámetro	Si (04 = 65) “Dispositivo a modificar”	01	Motor
			02	Otro
		Si (04 = 66) “Petición”	01	Recuperación
			02	Resguardo
			03	Reubicación
		Si (04 = 67)	00	NA
06	Segundo parámetro	Si (04 =65) “Parámetro a modificar”	01	Velocidad
			02	Otro

		Si (04 = 66) “Ubicación de la unidad”	XX		Cualquier entero
		Si (04= 67)	00		NA
07	Tercer Parámetro	Si (04 = 65) “Nuevo valor del Parámetro”	Si (05 = 1 && 06 = 1)	01	Lento
				02	Normal
				03	Rápido
				04	Máxima
		Si ((04 = 66 && (05 = 01||05=02))	00		NA
		Si (04 = 66 && 05=03) “Nueva Ubicación”	XX		Cualquier entero
		Si 04 = 67	00		NA
08	Cheksum		XX		Suma de los parámetros anteriores

Tabla 6: Codificación de la Información a utilizar

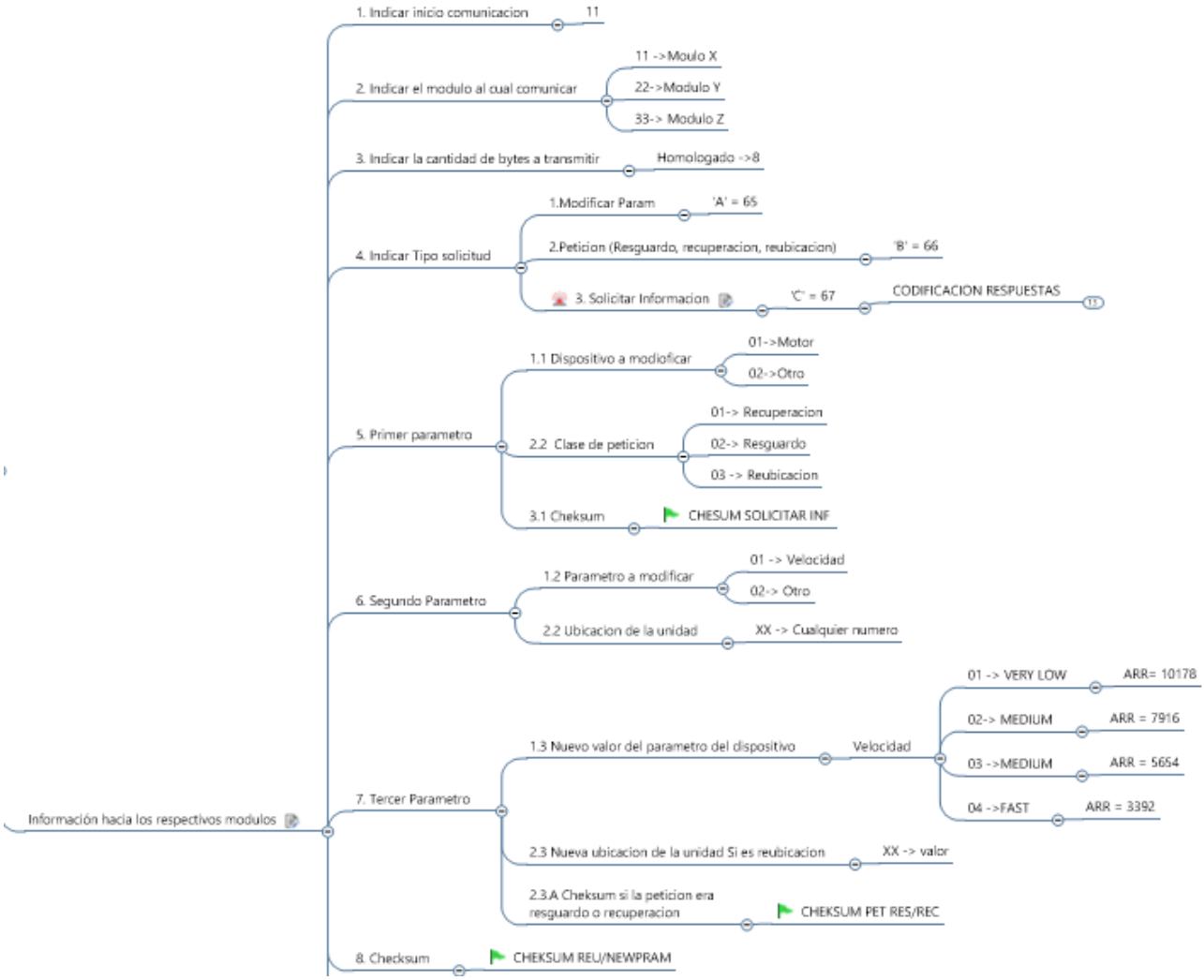


Figura 18: Diagrama de Comunicación 1

A continuación, se muestra formato de comunicación propuesto asi como el diagrama del mismo para la información que es enviada desde los microcontroladores, que mueven el robot hacia el microcontrolador que coordina el movimiento y comunica las órdenes a estos:

Byte	Nombre		Valor	Significado
01	Indicar inicio de comunicación		22	NA
02	Módulo que envía la información		11	Módulo X
			22	Módulo Y
			33	Módulo Z
03	Bytes a transmitir		9	NA
04	Temperatura microcontrolador		XX	Rango (-127 a 127)
05	Ubicación		XX	Coordenada (X||Y||Z)
06	Temperatura Lidar		XX	SI (02=33) -> NA
				Rango (-127 a 127)
07	Nivel Señal Lidar		XX	SI (02=33) -> NA
				Rango (0- 100)
08	Estado Motores	Si (02 = 11)	00	Motor X: OFF
			01	Motor Y: On
		Si (02 = 22)	00	Motor X: OFF
			01	Motor Y: On
		Si (02 = 33)	00	Todos los motores apagados
			01	Motor Telescópico: On
			02	Motores Extractor: On
			03	Todos los motores encendidos
09	Cheksum	XX		Suma de los parámetros anteriores

Tabla 7: Codificación de la Información

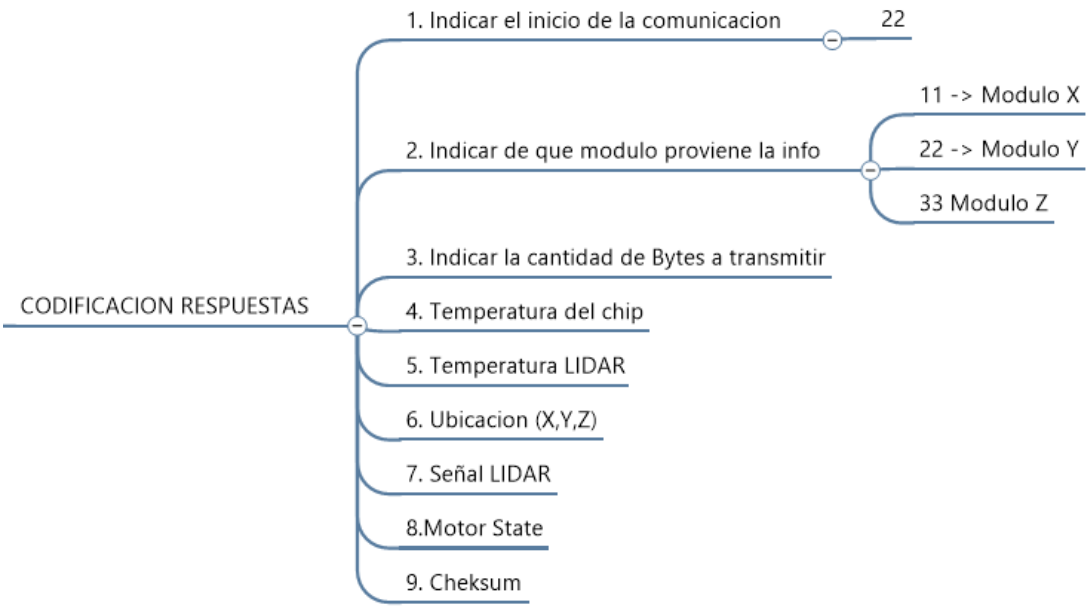


Figura 19: Diagrama de comunicación 2

A continuación, se muestra formato de comunicación propuesto, así como el diagrama del mismo para la información que es enviada desde el microcontrolador que controla el movimiento hacia los microcontroladores que mueven el robot para controlar la secuencia de movimiento:

Byte	Nombre	Valor	Significado
01	Encabezado	55	NA
02	Módulo al que va dirigida la orden	11	Módulo X
		22	Módulo Y
		33	Módulo Z
03	Bytes a transmitir	8	NA
04	Notificación	10	Inicialización del módulo principal: Ok
		20	Etapa 1: Completa
		30	Etapa 2: Completa
		40	Iniciar Etapa 1
		45	Iniciar Etapa 2
		50	Acknowledge (ACK)
05	Reservado	00	NA
06	Reservado	00	NA
07	Reservado	00	NA
08	Cerrar Encabezado	55	NA

Tabla 8: Codificación de la Información

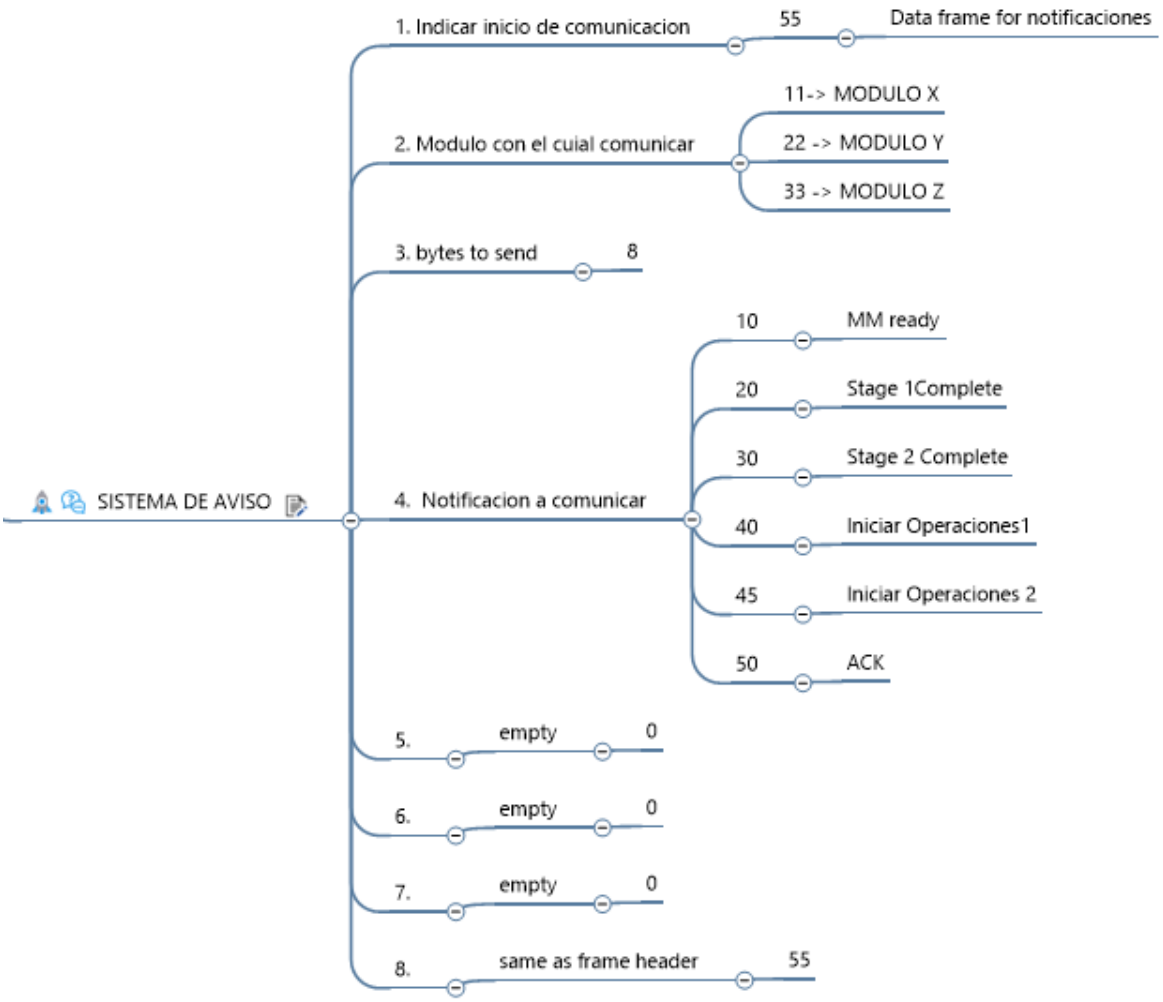


Figura 20: Diagrama de Comunicación 3

A continuación, se expone el formato utilizado para que el microcontrolador que controla la secuencia de movimiento del robot, así como la distribución de la información con los módulos responsables del movimiento en cada eje coordenado (formatos y diagramas expuestos previamente), reciba las Peticiones y los parámetros operativos que posteriormente tendrá que comunicar a cada módulo de movimiento mediante el formato expuesto en la tabla 7.

Byte	Nombre		Valor		Significado
01	Encabezado 1		25		NA
02	Encabezado 2		25		NA
03	Indicar la cantidad de bytes a transmitir		XX		El número de bytes a recibir es variable
04	Indicar el tipo de solicitud		01		Recuperación
			02		Resguardo
			03		Reubicación
			04		Modificar Parámetros
05	Primer parámetro	Si (04 = 01||02||03) “Ubicación Z”	XX		Actual Ubicación en el Eje Z de la Unidad
		Si (04 = 04) “Módulo”	11		Modificación para el Módulo X
			22		Modificación para el Módulo Y
			33		Modificación para el Módulo Z
06	Segundo parámetro	Si (04 = 01||02||03) “Ubicación X”	XX		Actual Ubicación en el Eje X de la Unidad
		Si (04 = 04) “Dispositivo a modificar”	01		Motor
			XX		Otro
07	Tercer Parámetro	Si (04 = 01||02||03) “Ubicación Y”	XX		Actual Ubicación en el Eje Y de la Unidad
		Si (04 = 04) “Parámetro a modificar”	Si (06=01)	01	Modificar Velocidad
			XX		Otro
08	Cuarto Parámetro	Si (04 = 01||02) “ID Unidad: Byte 1”	XX		ID Unidad
		Si (04 = 3) “Nueva Ubicación Z”	XX		Nueva Ubicación “Z” para la Unidad
		Si (04 = 04) “Nuevo valor del Parámetro”	XX		Nuevo valor del parámetro
09	Quinto Parámetro	Si (04 = 01||02) “ID Unidad: Byte 2”	XX		ID Unidad
		Si (04 = 3) “Nueva Ubicación X”	XX		Nueva Ubicación “X” para la Unidad
		Si (04 = 04) “CHEKSUM”	XX		Sumatoria
10	Sexto Parámetro	Si (04 = 01||02) “ID Unidad: Byte 3”	XX		ID Unidad
		Si (04 = 3) “Nueva Ubicación Y”	XX		Nueva Ubicación “Y” para la Unidad
11	Séptimo Parámetro	Si (04 = 01||02) “ID Unidad: Byte 4”	XX		ID Unidad

		Si (04 = 3) “ID Unidad: Byte 4”	XX	ID Unidad
12	Octavo Parámetro	Si (04 = 01||02) “CHEKSUM”	XX	Sumatoria
		Si (04 = 3) “ID Unidad: Byte 2”	XX	ID Unidad
13	Noveno Parámetro	Si (04 = 3) “ID Unidad: Byte 3”	XX	ID Unidad
14	Decimo Parámetro	Si (04 = 3) “ID Unidad: Byte 4”	XX	ID Unidad
15	Undécimo Parámetro	Si (04 = 3) “CHEKSUM”	XX	Sumatoria

Tabla 9: Codificación de la Información

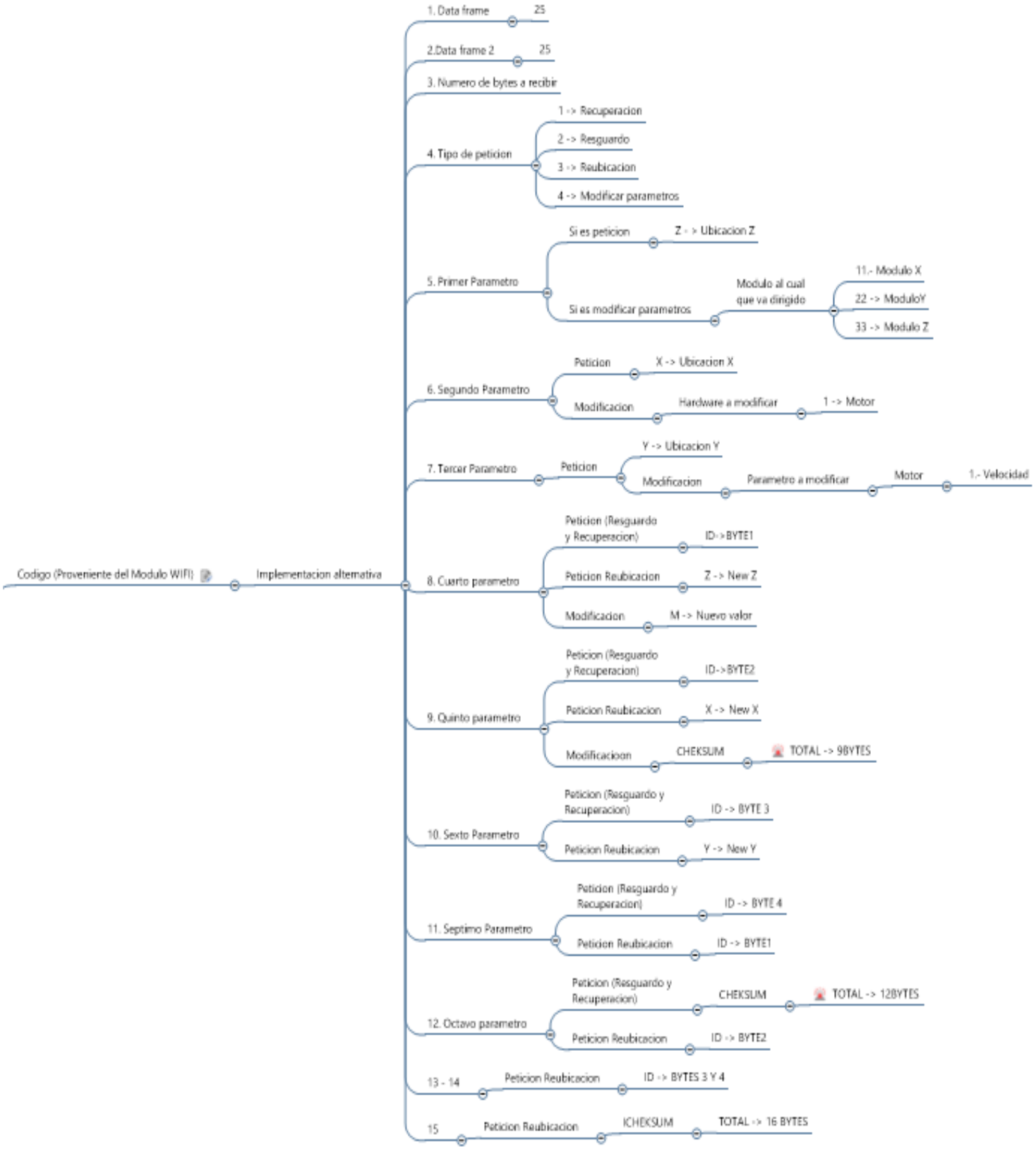


Figura 21: Diagrama de Comunicación 4

8.8 Paradigma de Funcionamiento

A continuación, se expone el modo de operación general del prototipo:

Todo debe comenzar con la petición de un usuario de un libro o archivo resguardado en un contenedor o unidad logística, que a su vez se encuentra en una posición determinada dentro de un rack que contiene otras unidades logísticas con más materiales bibliográficos dentro de estas.

La petición es enviada al prototipo a través de internet o de una conexión de área local, el prototipo recibe la petición y atiende a ella, moviéndose a las coordenadas donde se encuentra la unidad de resguardo (X, Y, Z) una vez en posición el robot procede a accionar el mecanismo de extracción de unidades de resguardo con el que está equipado, toma la unidad y la lleva al punto de entrega, una vez que la entrega se completó, el prototipo va a su posición HOME y espera una nueva petición, ya sea de entrega o de resguardo de una unidad entregada previamente.

A continuación, se muestra la secuencia de eventos que tienen que ocurrir para que el robot pueda iniciar o finalizar su operación, así como los parámetros esperados y comunicados a lo largo de la operación del robot.

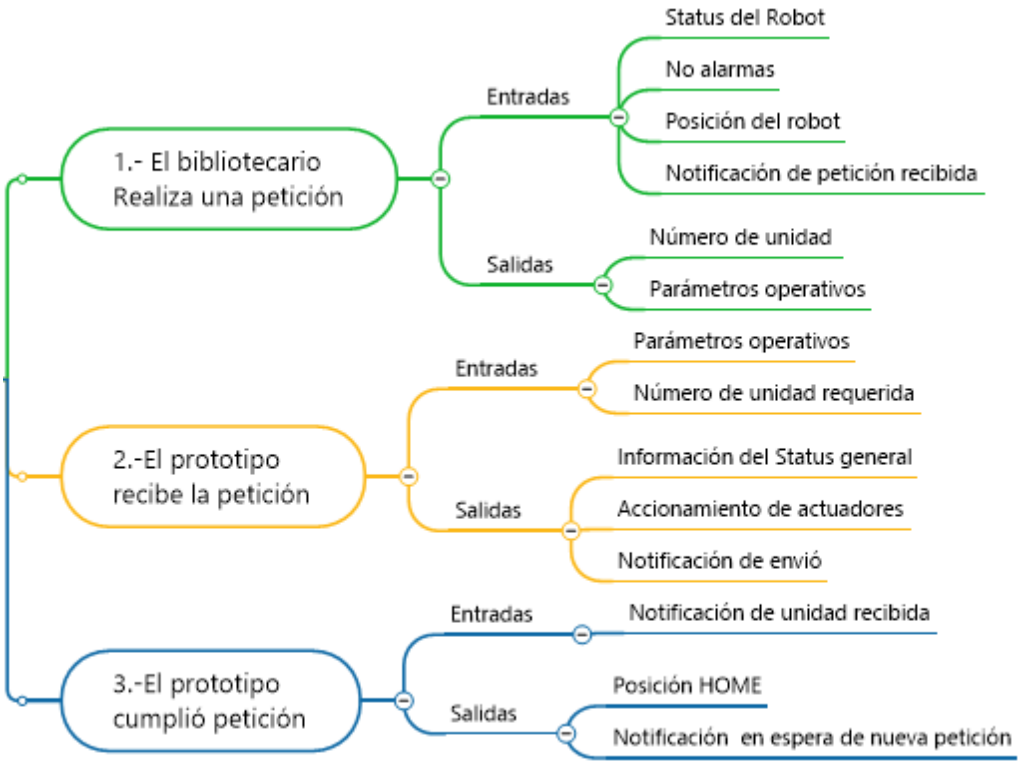


Figura 22: Diagrama del Paradigma de funcionamiento general.

A continuación, una explicación más detallada del paradigma de funcionamiento, así como de las etapas medulares de operación:

8.9 Inicialización del Sistema:

En esta etapa cada módulo inicializa los parámetros operativos, banderas en el software, se inicializan los drivers y la configuración inicial de cada microcontrolador responsable de cada módulo de operación.

Cada módulo entra en modo de espera hasta que finalicen de modo exitoso la inicialización completa todos y cada uno de los módulos. Cuando el módulo Y, X y Z han finalizado su inicialización, envían la correspondiente señal al módulo principal el cual responde a cada uno que ha sido recibido exitosamente su mensaje, para posteriormente entrar en modo de espera hasta recibir una orden.
El módulo principal, responsable de la secuencia de movimiento y sincronización del robot, se queda monitoreando la llegada de Peticiones del usuario, además de recibir el estado de cada uno de los módulos y publicándolo a la interfaz de usuario.

8.10 Petición de Usuario:

Cuando el módulo principal finalmente recibe una petición de usuario, este comunica con los módulos X, Y, Z, la información, todos reciben la misma información, pero cada uno reconoce los mensajes que van dirigidos a él, procesa el mensaje y procede a atender la petición del usuario que puede ser cualquiera de las siguientes:

8.10.1 Petición de Recuperación:

La recuperación tiene lugar cuando el usuario requiere realizar una consulta de algún material contenido en alguna unidad de resguardo. El usuario hace la petición por sí mismo o al bibliotecario, entonces la solicitud se procesa y es enviada al robot, el cual extrae la unidad del rack en el que se encuentra y la lleva a la zona de entrega para que inicie su camino hasta llegar con el usuario final.

8.10.2 Petición de Resguardo:

El resguardo tiene lugar cuando es necesario almacenar nuevamente la unidad que ya no es requerida, aunque hay otro momento importante en el cual la secuencia de resguardo tendrá lugar y ese es el acomodo inicial de las unidades de almacenaje en los racks correspondientes.

Inicia cuando desde la interfaz del Bibliotecario, se envía una solicitud de resguardo al robot, los sistemas de entrega llevan la unidad hasta los racks donde el robot la acomoda en el lugar que le corresponde en el rack acorde al número de unidad enviado junto con la solicitud de resguardo.

8.10.3 Petición de Reubicación:

La Petición de reubicación es utilizada para indicar al robot que debe mover una unidad a una nueva ubicación ya sea dentro del mismo rack o a otro diferente, para esto la información requerida es la ubicación actual, así como la nueva ubicación correspondiente a dicha unidad de resguardo.

8.10.4 Modificar Parámetros:

Es posible modificar parámetros operativos del robot, actualmente en esta primera iteración solo es posible modificar la velocidad de los motores dentro de un rango seguro de operación, lo que permite acortar los tiempos de operación del robot.

8.10.5 Secuencia de Cancelación

La secuencia de cancelación se lleva a cabo cuando por alguna razón no se requiere completar la solicitud o hubo un error en dicha solicitud, el robot cancela la operación y regresa la unidad al rack o nuevamente al sistema de entrega para que vuelva al Bibliotecario.

8.10.6 Gestión de Unidades y Base de datos

La gestión de unidades se lleva a cabo completamente por la interfaz del Bibliotecario, la cual se encarga de la identificación de todas las unidades y el contenido de estas, es decir tiene funciones de agregar/remover material bibliográfico o hemerográfico de las unidades de resguardo, cambio de material de unidad, reubicación de unidades y remoción de estas. Aquí también se verifica que solo se pueda procesar una petición de unidad de resguardo que se encuentre en los racks y no haya sido solicitada anteriormente.

8.11 Redundancia

La redundancia es un elemento primordial para alcanzar el desempeño esperado en un sistema complejo como lo es un ASRS, puede definirse como la duplicación de elementos críticos para mejorar la disponibilidad de un sistema [17].

En [17], se consideran dos tipos de redundancia que se resumen a continuación:

Redundancia Interna: La cual considera la redundancia referente al hardware y el software que son administradas por una misma instancia de un sistema.

Redundancia Externa: Considera un conjunto de sistemas que a su vez puede ser internamente redundante o no serlo, que se organizan en un grupo para ofrecer una mayor disponibilidad o capacidad a los usuarios finales u otros sistemas. Se puede dividir a la redundancia externa en dos categorías:

> *Redundancia Externa Co-Ubicada:* Cuando las instancias de sistemas redundantes están físicamente cerca unas de otras.
>
> *Redundancia Externa distribuida Geográficamente o "Geo-redundancia":* Cuando las instancias redundantes están físicamente separadas generalmente por grandes distancias, para minimizar los riesgos como por ejemplo factores naturales tales como inundaciones, terremotos, etc.

Para asegurar que el comportamiento del prototipo sea confiable y repetible se plantea la implementación de redundancia en varios de los subsistemas que lo componen. El tipo de redundancia implementada será básicamente redundancia interna y Redundancia Externa Co-Ubicada. Se describe a continuación los sistemas y subsistemas redundantes:

> ***Prototipo Físico:*** La importancia de la implementación de la redundancia es asegurar movimientos precisos y lograr un sistema de alarmas y notificaciones robusto.
> En el prototipo se ha incluido la redundancia en los siguientes sistemas:
>
> > Movimiento: Para iniciar el movimiento el robot primeramente calcula la distancia al objetivo mediante el sensor Lidar, con la distancia obtenida el microcontrolador del respectivo módulo calcula los parámetros operativos del motor, acto seguido el motor se enciende e inicia el movimiento, en tanto el sensor liDAR sigue monitoreando el avance del prototipo en el eje coordenado, por lo que si el motor no se detiene por alguna razón desconocida, el microcontrolador puede realizar una segunda verificación mediante la lectura del sensor. Adicional a esto se han incorporado sensores de final de carrera en el prototipo para evitar avances más allá de los límites físicos permitidos.

Intercambio de información: Cada mensaje en el prototipo posee al menos dos bytes de identificación: el encabezado y el Cheksum. El encabezado es utilizado para verificar que el destino del mensaje es el correcto y finalmente se verifica el Cheksum para asegurar que el mensaje ha sido recibido íntegramente y que el mensaje no ha sido corrompido por ruido eléctrico o algún otro fenómeno no deseado. La verificación del Cheksum se realiza dos veces, una por quien envía el mensaje y otra por quien lo recibe, adicional a esto, hay parámetros de verificación para asegurar que los parámetros recibidos están dentro del rango de operación del robot y suponen un movimiento seguro, esta comprobación es realizada en cada microcontrolador cuando recibe un mensaje.

Registro de Operaciones: El prototipo lleva un registro de las operaciones realizadas en cada momento, almacena el ID de la unidad que almacenó, entró o reubicó, registro que también será almacenado en la interfaz gráfica del usuario.
Adicional al registro de las operaciones realizadas, también existirá un registro de alarmas y códigos de error ocurridos durante la operación del mismo.

9 CONSTRUCCIÓN O IMPLEMENTACIÓN DEL PROTOTIPO

La etapa de construcción del prototipo fue quizá la más extensa junto a la etapa de diseño, representó importantes desafíos y causo varias iteraciones en el diseño inicial del prototipó.

En especial el desarrollo de las tarjetas de control, al estar hechas por métodos “caseros” de fabricación y también a mi falta de experiencia en el desarrollo de electrónica fue la parte que sin duda requirió mucho retrabajo.

Por mencionar parte del retrabajo que se realizó en la parte electrónica, se tuvo que cambiar el diseño debido a una mala elección de convertidores de nivel de voltaje TTL, utilizados para el acoplamiento de señales de sensores, señales de control de los servomotores asi como de las comunicaciones, las cuales utilizan un voltaje de 5 volts, pero el microcontrolador trabaja con 3.3 volts.

El cambio se realizó debido a que la salida que estos producen, es muy sensible a la capacitancia de la tarjeta, a la distancia desde la cual se envía la señal de entrada y la de salida y a otros fenómenos eléctricos.

Figura 23: Materiales de Construcción

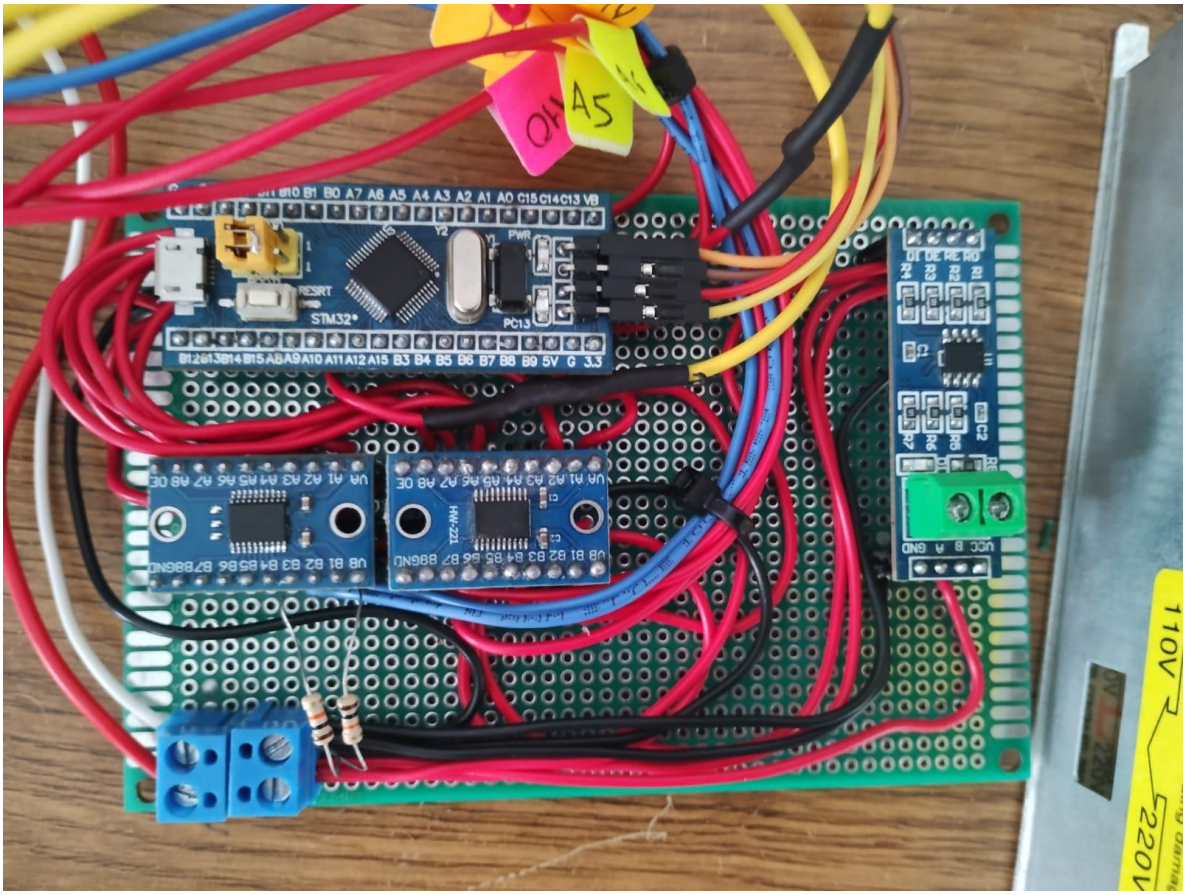


Figura 24: Tarjeta que tuvo que ser rediseñada

En el lugar de los circuitos integrados convertidores de voltaje se utilizaron MOSFETS BSS138, a continuación, se muestra el arreglo utilizado:

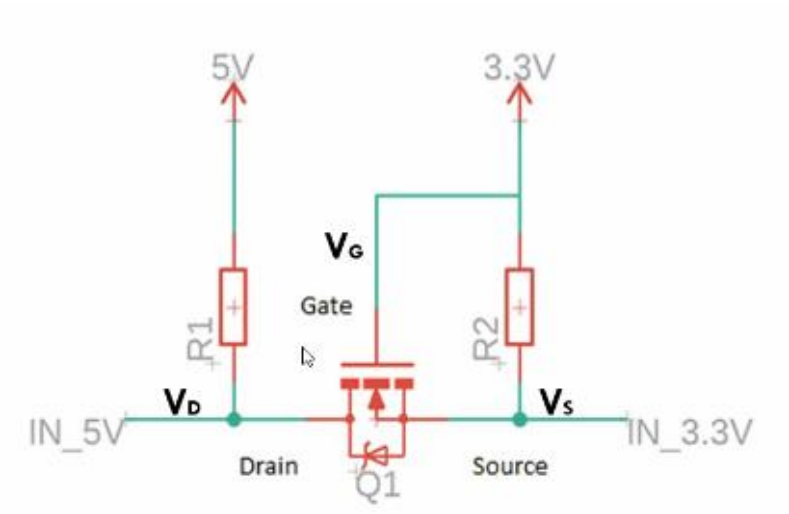


Figura 25: Diagrama Electrónico del Convertidor de voltaje TTL usando MOSFETS

Como se aprecia en las imágenes el prototipo lo construí en casa, con herramientas de mano y materiales al alcance de mi economía. Entre los que destacan:

- Perfil de aluminio extruido en diferentes dimensiones: 20*20 mm y 20 *40mm
- MDF de 5 mm de espesor
- Lamina de acero para algunas uniones
- Tornillería [5mm,4mm,3mm]
- Cable calibre 20 en diferentes colores
- Bandas HTD3 15mm de ancho.

Se utilizaron técnicas de manufactura aditiva como la impresión 3d para generar algunas piezas importantes del prototipo como poleas, soportes, el piñón y cremallera del mecanismo telescópico entre otros.

Figura 26: Vista de algunas Piezas producidas en impresión 3d

El cableado e instalación de las tarjetas se realizó de modo que estuvieran lo más cerca posible del sensor LiDAR para evitar problemas con la transferencia de información, por lo cual la tarjeta del Módulo que controla el movimiento en el eje Y, se ha instalado en la base en la que también se instala el extractor, esta base realiza el movimiento ascendente y descendente en el eje coordenado “Y”.

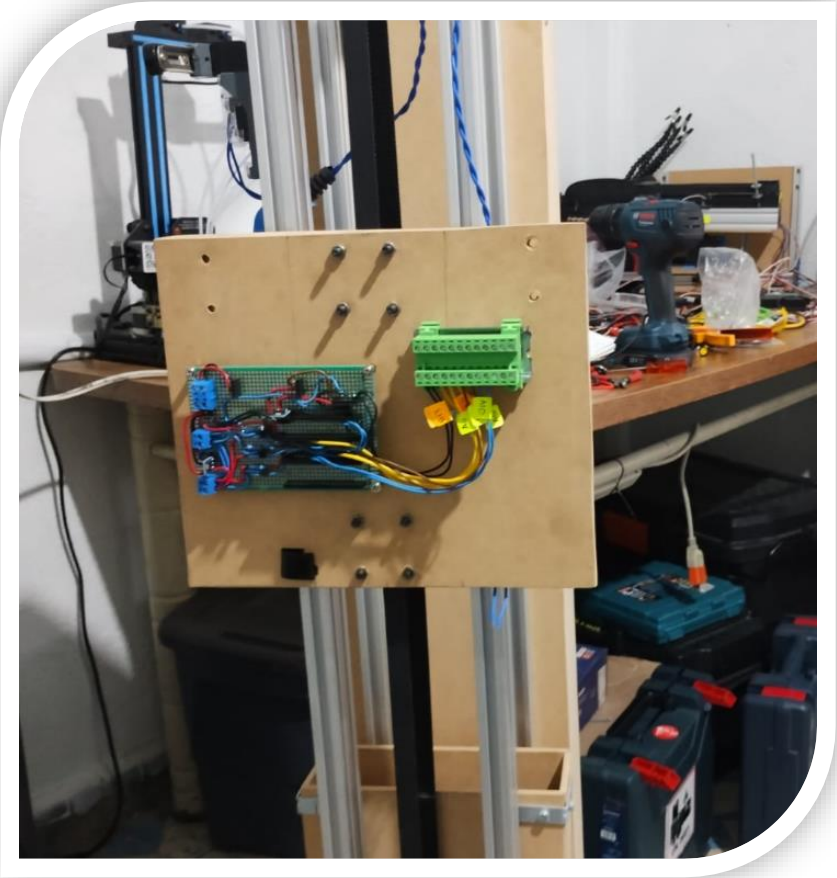


Figura 27: Instalación del módulo de control del eje Y

En el gabinete se han instalado las tarjetas de control de los módulos que controlan el movimiento del mecanismo extractor en el eje coordenado “Z” asi como el avance del robot en el eje coordenado “X”. También en el gabinete se han instalado los drivers de control de cada uno de los servomotores y motores a pasos que usa el prototipo, dos fuentes de alimentación de 5V, la cuales alimenta cada uno de los microcontroladores, los sensores Lidar, los sensores de final de carrera, el acelerómetro entre otros.

La tarjeta de control que controla la secuencia de movimiento de los ejes coordenados y de comunicación, la fuente de alimentación para los servomotores y motores a pasos también han sido instaladas en el gabinete.

Figura 28: Vista del gabinete siendo cableado

Figura 29: Vista del gabinete finalizado

Figura 30: Prototipo finalizado

10 REALIZACIÓN DE PRUEBAS Y ADQUISICIÓN DE DATOS

Se llevaron a cabo extensas sesiones de prueba del funcionamiento de cada módulo antes de realizar el ensamblaje final, se comprobó cada línea de código usando el depurador o "debugger" de la compañía STM32 "STM Cube IDE", el intercambio de información entre cada módulo resulto ser el esperado. A continuación, se detallan las pruebas hechas para cada módulo del prototipo:

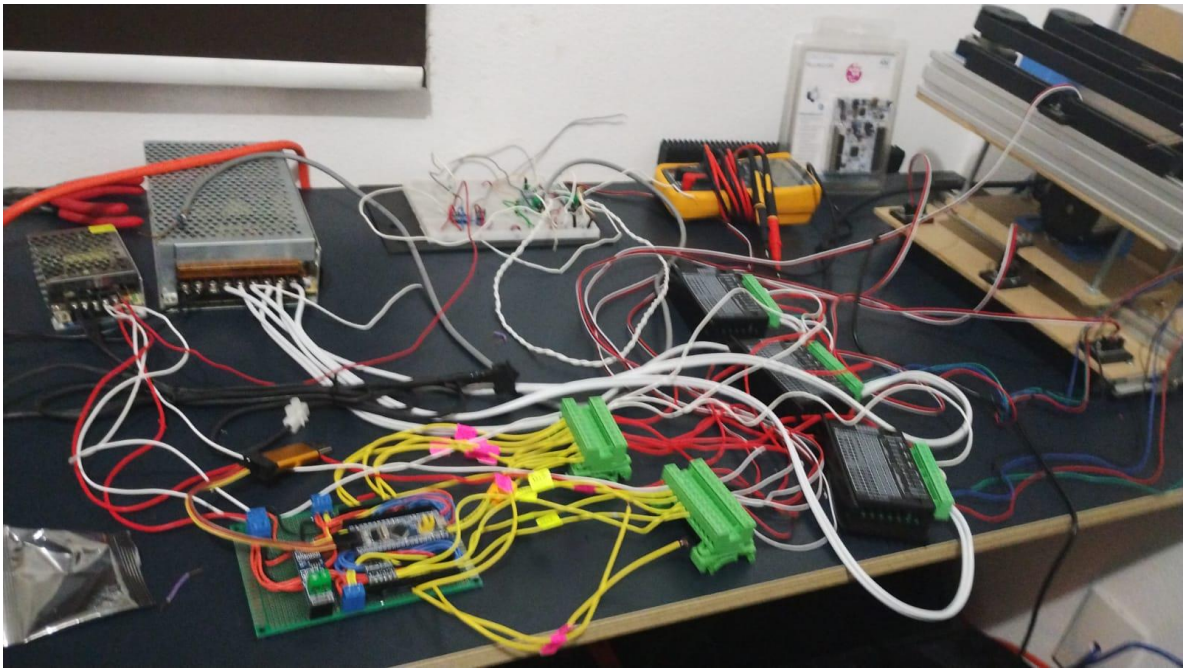


Figura 31: Módulo Z en pruebas de funcionamiento.

10.1 Test Cumplimiento de la Secuencia de Recuperación:

A continuación, se muestran los resultados de una de las pruebas realizadas para comprobar la secuencia de "Recuperación". Los parámetros utilizados fueron los siguientes:

COORDENADA		
X	***Y***	***Z***
5	*3*	*1*

Tabla 10: Coordenadas seleccionadas para el test (recuperación) en cada eje de movimiento

Notificación de Inicialización completa del módulo principal			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 0A 00 00 00 37	37 16 08 0A 00 00 00 37	37 21 08 0A 00 00 00 37
Comportamiento Esperado	El módulo debe responder encendiendo uno de sus Pines para indicar al módulo principal que ha recibido el mensaje, acto seguido entra en modo de espera, hasta recibir los datos de una petición de usuario. El módulo principal entonces envía el ACK para indicar al módulo X que sabe que recibió el mensaje.		
¿Se Cumple el comportamiento esperado?	Si, el módulo enciende el Pin correctamente al recibir este mensaje y espera el ACK del módulo principal.		
ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 11: Test de inicialización

Datos de la petición realizada por el usuario			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	0B 0B 08 42 01 05 00 66	0B 16 08 42 01 03 00 6F	0B 21 08 42 01 01 00 78
Comportamiento Esperado	Los módulos deben recibir la información correcta y correspondiente para cada uno de ellos, después de procesar la información deben avisar al módulo principal que han recibido la información correctamente y entrar en modo de espera hasta que el módulo principal envié el respecto mensaje de ACK a cada uno de ellos.		
¿Se Cumple el comportamiento esperado?	Si, el módulo enciende el Pin correctamente al recibir este mensaje y entra en modo de espera del ACK del módulo principal.		

ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 12: Test recepción de peticiones

Notificación de inicio de la Primera fase de movimiento			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 28 00 00 00 37	37 16 08 28 00 00 00 37	37 21 08 28 00 00 00 37
Comportamiento Esperado	Al recibir esta orden cada módulo debe dar inicio al primer parte de su respectiva secuencia de movimiento.		
¿Se Cumple el comportamiento esperado?	Si, cada módulo realiza su secuencia de movimiento respectiva en tiempo y forma.		
ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 13: Test primera secuencia de movimiento completa

Reporte del estado de la primera fase de movimiento			
	Módulo X	Módulo Y	Módulo Z

Trama de datos del Mensaje [Hex]	37 0B 08 14 00 00 00 37	37 16 08 14 00 00 00 37	37 21 08 14 00 00 00 37
Comportamiento Esperado	Cada módulo debe indicar al módulo principal su estado actual, si aún se encuentra realizando movimiento o a finalizado la orden anterior.		
¿Se Cumple el comportamiento esperado?	Si, cada módulo indica de manera correcta si aún se encuentra en funcionamiento o ha terminado su recorrido.		
ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 14: Test reporte estado después de la primera fase de movimiento

Notificación de inicio de la segunda fase de movimiento			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 2D 00 00 00 37	37 16 08 2D 00 00 00 37	37 21 08 2D 00 00 00 37
Comportamiento Esperado	Al recibir esta orden cada módulo debe dar inicio a la última parte de su respectiva secuencia de movimiento.		
¿Se Cumple el comportamiento esperado?	Si, cada módulo realiza su secuencia de movimiento respectiva en tiempo y forma.		
ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 15: Test segunda fase de movimiento

Reporte del estado de la segunda fase de movimiento			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 1E 00 00 00 37	37 16 08 1E 00 00 00 37	37 21 08 1E 00 00 00 37
Comportamiento Esperado	Cada módulo debe indicar al módulo principal su estado actual, si aún se encuentra realizando movimiento o a finalizado la orden anterior.		
¿Se Cumple el comportamiento esperado?	Si, cada módulo indica de manera correcta si aún se encuentra en funcionamiento o ha terminado su recorrido.		
ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 16: Test reporte estado después de la segunda fase de movimiento

ACK Final			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
Comportamiento Esperado	Cada módulo debe reiniciar sus banderas y realizar todas las secuencias requeridas para poder atender nuevamente una petición, como ir a home.		
¿Se Cumple el comportamiento esperado?	Si, los módulos se ponen a punto para la siguiente petición de modo correcto.		

Tabla 17: Test finalización de operaciones

10.1.1 Test Cumplimiento de la Secuencia de Resguardo

A continuación, se muestran los resultados de una de las pruebas realizadas para comprobar la secuencia de "Resguardo". Los parámetros utilizados fueron los siguientes:

COORDENADA		
X	***Y***	***Z***
5	*3*	*2*

Tabla 18: Coordenadas seleccionadas para el test (resguardo) en cada eje de movimiento

Notificación de Inicialización completa del módulo principal			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 0A 00 00 00 37	37 16 08 0A 00 00 00 37	37 21 08 0A 00 00 00 37
Comportamiento Esperado	El módulo debe responder encendiendo uno de sus Pines para indicar al módulo principal que ha recibido el mensaje, acto seguido entra en modo de espera, hasta recibir los datos de una petición de usuario. El módulo principal entonces envía el ACK para indicar al módulo X que sabe que recibió el mensaje.		
¿Se Cumple el comportamiento esperado?	Si, el módulo enciende el Pin correctamente al recibir este mensaje y espera el ACK del módulo principal.		
ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 19: Test de inicialización

Datos de la petición realizada por el usuario			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	0B 0B 08 42 02 05 00 66	0B 16 08 42 02 03 00 6F	0B 21 08 42 02 02 00 78
Comportamiento Esperado	Los módulos deben recibir la información correcta y correspondiente para cada uno de ellos, después de procesar la información deben avisar al módulo principal que han recibido la información correctamente y entrar en modo de espera hasta que el módulo principal envié el respecto mensaje de ACK a cada uno de ellos.		
¿Se Cumple el comportamiento esperado?	Si, el módulo enciende el Pin correctamente al recibir este mensaje y entra en modo de espera del ACK del módulo principal.		
ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 20: Test recepción de peticiones

Notificación de inicio de la Primera fase de movimiento			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 28 00 00 00 37	37 16 08 28 00 00 00 37	37 21 08 28 00 00 00 37
Comportamiento Esperado	Al recibir esta orden cada módulo debe dar inicio al primer parte de su respectiva secuencia de movimiento.		
¿Se Cumple el comportamiento esperado?	Si, cada módulo realiza su secuencia de movimiento respectiva en tiempo y forma.		
	Módulo X	Módulo Y	Módulo Z

ACK esperado por el módulo [Hex]	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 21: Test primera secuencia de movimiento completa

Reporte del estado de la primera fase de movimiento			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 14 00 00 00 37	37 16 08 14 00 00 00 37	37 21 08 14 00 00 00 37
Comportamiento Esperado	Cada módulo debe indicar al módulo principal su estado actual, si aún se encuentra realizando movimiento o a finalizado la orden anterior.		
¿Se Cumple el comportamiento esperado?	Si, cada módulo indica de manera correcta si aún se encuentra en funcionamiento o ha terminado su recorrido.		
ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 22: Test reporte estado después de la primera fase de movimiento

Notificación de inicio de la segunda fase de movimiento			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 2D 00 00 00 37	37 16 08 2D 00 00 00 37	37 21 08 2D 00 00 00 37

Comportamiento Esperado	Al recibir esta orden cada módulo debe dar inicio a la última parte de su respectiva secuencia de movimiento.		
¿Se Cumple el comportamiento esperado?	Si, cada módulo realiza su secuencia de movimiento respectiva en tiempo y forma.		
ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 23: Test segunda fase de movimiento

Reporte del estado de la segunda fase de movimiento			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 1E 00 00 00 37	37 16 08 1E 00 00 00 37	37 21 08 1E 00 00 00 37
Comportamiento Esperado	Cada módulo debe indicar al módulo principal su estado actual, si aún se encuentra realizando movimiento o a finalizado la orden anterior.		
¿Se Cumple el comportamiento esperado?	Si, cada módulo indica de manera correcta si aún se encuentra en funcionamiento o ha terminado su recorrido.		
ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 24: Test reporte estado después de la segunda fase de movimiento

ACK Final			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
Comportamiento Esperado	Cada módulo debe reiniciar sus banderas y realizar todas las secuencias requeridas para poder atender nuevamente una petición, como ir a home.		
¿Se Cumple el comportamiento esperado?	Si, los módulos se ponen a punto para la siguiente petición de modo correcto.		

Tabla 25: Test finalización de operaciones

10.2 Test Cumplimiento de la Secuencia de Reubicación

A continuación, se muestran los resultados de una de las pruebas realizadas para comprobar la secuencia de "Reubicación". Los parámetros utilizados fueron los siguientes:

COORDENADAS INICIALES		
X	***Y***	***Z***
5	*3*	*2*
COORDENADAS FINALES		
X	***Y***	***Z***
2	*1*	*1*

Tabla 26: Coordenadas seleccionadas para el test (reubicación) en cada eje de movimiento

Notificación de Inicialización completa del módulo principal			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 0A 00 00 00 37	37 16 08 0A 00 00 00 37	37 21 08 0A 00 00 00 37
Comportamiento Esperado	El módulo debe responder encendiendo uno de sus Pines para indicar al módulo principal que ha recibido el mensaje, acto seguido entra en modo de espera, hasta recibir los datos de una petición de usuario. El módulo principal entonces envía el ACK para indicar al módulo X que sabe que recibió el mensaje.		

¿Se Cumple el comportamiento esperado?	Si, el módulo enciende el Pin correctamente al recibir este mensaje y espera el ACK del módulo principal.		
ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 27: Test de inicialización

Datos de la petición realizada por el usuario			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	0B 0B 08 42 03 05 02 70	0B 16 08 42 03 03 01 72	0B 21 08 42 03 02 01 7C
Comportamiento Esperado	Los módulos deben recibir la información correcta y correspondiente para cada uno de ellos, después de procesar la información deben avisar al módulo principal que han recibido la información correctamente y entrar en modo de espera hasta que el módulo principal envié el respecto mensaje de ACK a cada uno de ellos.		
¿Se Cumple el comportamiento esperado?	Si, el módulo enciende el Pin correctamente al recibir este mensaje y entra en modo de espera del ACK del módulo principal.		
ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 28: Test recepción de peticiones

Notificación de inicio de la Primera fase de movimiento			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 28 00 00 00 37	37 16 08 28 00 00 00 37	37 21 08 28 00 00 00 37
Comportamiento Esperado	Al recibir esta orden cada módulo debe dar inicio al primer parte de su respectiva secuencia de movimiento.		
¿Se Cumple el comportamiento esperado?	Si, cada módulo realiza su secuencia de movimiento respectiva en tiempo y forma.		
ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 29: Test primera secuencia de movimiento completa

Reporte del estado de la primera fase de movimiento			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 14 00 00 00 37	37 16 08 14 00 00 00 37	37 21 08 14 00 00 00 37
Comportamiento Esperado	Cada módulo debe indicar al módulo principal su estado actual, si aún se encuentra realizando movimiento o a finalizado la orden anterior.		
¿Se Cumple el comportamiento esperado?	Si, cada módulo indica de manera correcta si aún se encuentra en funcionamiento o ha terminado su recorrido.		
ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 30: Test reporte estado después de la primera fase de movimiento

Notificación de inicio de la segunda fase de movimiento			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 2D 00 00 00 37	37 16 08 2D 00 00 00 37	37 21 08 2D 00 00 00 37
Comportamiento Esperado	Al recibir esta orden cada módulo debe dar inicio a la última parte de su respectiva secuencia de movimiento.		
¿Se Cumple el comportamiento esperado?	Si, cada módulo realiza su secuencia de movimiento respectiva en tiempo y forma.		
ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 31: Test segunda fase de movimiento

Reporte del estado de la segunda fase de movimiento			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 1E 00 00 00 37	37 16 08 1E 00 00 00 37	37 21 08 1E 00 00 00 37
Comportamiento Esperado	Cada módulo debe indicar al módulo principal su estado actual, si aún se encuentra realizando movimiento o a finalizado la orden anterior.		
¿Se Cumple el comportamiento esperado?	Si, cada módulo indica de manera correcta si aún se encuentra en funcionamiento o ha terminado su recorrido.		

ACK esperado por el módulo [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
ACK recibido	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37

Tabla 32: Test reporte estado después de la segunda fase de movimiento

ACK Final			
Trama de datos del Mensaje [Hex]	Módulo X	Módulo Y	Módulo Z
	37 0B 08 32 00 00 00 37	37 16 08 32 00 00 00 37	37 21 08 32 00 00 00 37
Comportamiento Esperado	Cada módulo debe reiniciar sus banderas y realizar todas las secuencias requeridas para poder atender nuevamente una petición, como ir a home.		
¿Se Cumple el comportamiento esperado?	Si, los módulos se ponen a punto para la siguiente petición de modo correcto.		

Tabla 33: Test finalización de operaciones

11 DOCUMENTACIÓN Y REPORTE DE RESULTADOS

Dividiré el reporte de resultados en lo que considero los pilares fundamentales para la realización de un proyecto de esta magnitud: Software, Hardware, Diseño Mecánico, Ingeniería de Sistemas.

11.1 Software:

El software fue sin duda lo que mejor salió de este proyecto y menos retrabajo necesito. Aunque sin duda hay muchas cosas que mejorar algunas de las cuales he citado están planeadas como trabajo futuro, entre las que puedo destacar en este apartado implementar un manejador de errores o "error handler" cuyo nombre en inglés hace mejor referencia al propósito de este.

Todos y cada uno de los drivers utilizados en la industria automotriz y aeroespacial, cuentan con *error handlers*, cuyo propósito es identificar errores ya sea en la transmisión de datos, en la lectura de alguna señal eléctrica, variaciones no esperadas en algún pin del microcontrolador, etc., todos los posibles escenarios son considerados y se escribe código para atender cada una de las posibilidades, y almacenar el suceso como un código de falla.

Durante las pruebas presencie algunos errores que me hicieron gastar horas investigando que es lo que había pasado y porque es que había sucedido, por mencionar algunos, el pobre diseño electrónico implementado para hacer funcionar cada uno de los microcontroladores del prototipo, asi como las demás fuentes de ruido que siempre suelen estar presentes en el entorno ocasionaron algunos errores como son:

- Corrupción de la información enviada a través del BUS RS485
- Falsas lecturas de los diversos sensores utilizados
- Errores de escritura en los registros del microcontrolador

Si bien los errores anteriormente mencionados hubieran seguido suscitándose ya que todos eran ocasionados por errores de hardware, un buen manejador de errores hubiera sido de gran ayuda para saber qué es lo que estaba sucediendo y en que preciso momento es que ocurrían los errores. Además de tomar acciones preventivas para evitar comportamientos indeseados en el prototipo y establecer un marco de comportamiento "seguro" ante la presencia de errores inesperados, lo que nos lleva a una necesidad adicional: Implementar un sistema operativo en tiempo real o RTOS "Real Time Operating System" por sus siglas en ingles.

Ya que derivado de los comportamientos indeseados, que si bien fueron causados en su momento por el ruido electrónico me hicieron dar cuenta de la necesidad

fundamental de implementar un RTOS, cuando ocurría algún error en la lectura de los sensores LiDAR, es decir alguna lectura fuera de rango, el robot por seguridad no avanzaba del punto en el que los datos no eran consistentes con constantes programadas para evitar movimientos fuera de rango, es decir si en algún momento por alguna razón el sensor deja de funcionar o enviar datos correctos, el robot podría quedarse atrapado en un bucle, y aunque se escribiera un manejador de errores, podría no ejecutarse debido a que el robot está atrapado en un bucle dentro de una función en donde el manejador de errores no es llamado, es aquí donde entra el sistema operativo, el cual se encargara de manejar los recursos del microcontrolador en su totalidad, controlando el tiempo de ejecución de cada tarea, asi como los recursos de procesamiento que cada tarea puede usar y por cuanto tiempo, de este modo aunque el microcontrolador usado no tenga capacidades multicore, el procesador podría ejecutar múltiples tareas una después de otra cuando cada una ha agotado el tiempo de uso del procesador, o si se selecciona un procesador con capacidades multicore, entonces tareas pueden ser ejecutadas en paralelo o al mismo tiempo, por lo cual al estarse ejecutando el manejador de errores como una tarea periódica, podría darse cuenta de que el sensor LiDAR está enviando datos fuera de rango y registrar el código de falla, lo que ahorraría demasiado tiempo en encontrar el problema por el cual el robot no funciona.
Y sobre todo evitaría que el robot quedara atrapado en algún punto de la ejecución del software por cualquier otro motivo sin razón aparente, o al menos si queda atrapado podría el sistema operativo registrar el evento en un código de falla.

11.2 Hardware:

Esto fue sin duda alguna lo que más complicaciones presentó al proyecto, algunas ya mencionadas en el apartado anterior como lecturas inadecuadas, corrupción de la información, etc.

Con la ayuda de un osciloscopio fue posible determinar las fuentes de ruido en el sistema lo cual llevo a las soluciones implementadas:

- Se colocaron capacitores cerámicos entre los pines de alimentación del microcontrolador.
- Se cambiaron algunos circuitos integrados utilizados para convertir voltajes ttl (3.3 a 5v y viceversa) por mosfets.
- Se redujo la distancia de los sensores LiDAR al microcontrolador.
- Se configuro cada pin no usado del microcontrolador como una salida y se puso a nivel bajo por software (0v)

- Se modificó la secuencia de encendido de los componentes del prototipo, primero encienden todos los motores y después los microcontroladores.

Una vez solucionados los problemas de hardware, el funcionamiento en conjunto con el software fue el planeado en la ingeniera de sistemas, el cual ha sido presentado a lo largo de esta tesis, los mensajes enviados y recibidos por cada módulo fueron los esperados, el tiempo de traslado en cada eje asi como las secuencias que tendrían que suceder a cada evento en el sistema.

El hardware también está planeado como una importante mejora en el trabajo a futuro, se planea desarrollar el esquemático de cada tarjeta para mandar a fabricar de modo profesional la tarjeta electrónica de cada módulo.

De aquí destacaría sin duda la importancia de un buen diseño electrónico.

Otro percance suscitado durante las pruebas fue el sobrecalentamiento de uno de los motores del extractor debido a una mala configuración en el driver, lo que planteo la necesidad de incluir monitoreo para el hardware primordial del sistema como lo son los motores, sensores etc.

Derivado de esto se ha decidido incluir en el trabajo futuro, un sistema de monitoreo de consumo de corriente, voltaje y temperatura de los motores para evitar sobrecarga y detectar consumos anormales de corriente antes de quemar el motor, lo cual brindaría ayuda también a realizar mantenimiento predictivo al sistema.

Esta solución se extendería a sensores y hardware costoso y relevante.

11.3 Diseño Mecánico

El diseño mecánico sin duda fue de las cosas que más disfruté, junto al desarrollo del software. Debido a la naturaleza del proyecto, el prototipo era una hoja en blanco cuya única restricción era la imaginación y claro el presupuesto.

Ya que deseaba que el proyecto fuera lo más representativo que pudiera con mis medios, fue el presupuesto la variable más importante en la toma de decisiones, como lo fueron el tamaño, los componentes a utilizar y las prestaciones del prototipo.

Ya que los ASRS de columna son soluciones que han probado ser eficientes y tener de los mejores desempeños en la industria, es por eso que para una primera iteración fue el concepto elegido.

Para el extractor si bien existían nuevamente diferentes maneras de realizar el trabajo, el extractor de cadena sin duda es de los mecanismos más sencillos, que aprovechan al máximo el espacio, aunque aumentan los costos iniciales al requerir unidades de almacenaje especialmente diseñadas para poder utilizar este

mecanismo, aunque sin duda quizá el mayor aprovechamiento de espacio compense el mayor costo inicial de fabricación.

Además, ya que la intención es probar diferentes soluciones, el extractor puede ser remplazado por alguno que implemente otro mecanismo para extraer las unidades como ventosas, levantar la unidad desde la base, o alguna otra solución podría rediseñarse un nuevo extractor y adaptarse a la columna existente.

Analizando el comportamiento del prototipo en movimiento y reposo, sin duda alguna los soportes de la columna son los elementos que requieren más atención, ya que soportaran la carga de aceleración y desaceleración del conjunto tren motriz – columna -extractor.

El mecanismo extractor puede ser mejorado para utilizar un solo motor, y mediante una transmisión conseguir el accionamiento de ambos pines.

Es necesario incorporar un tensor de cadena, para disminuir el ruido y evitar movimientos errantes de la cadena, también si la distancia entre ejes se aumenta será necesario agregar guías para que la cadena no vibre ni tampoco se cuelgue.

También es importante implementar mejoras en la interacción entre el extractor y la columna, con el diseño actual, dos placas de desplazamiento son utilizadas para mover el mecanismo extractor a lo largo de la columna. Si bien han mostrado ser eficaces cuando el prototipo está en movimiento ascendente o descendente, es evidente una ligera inclinación cuando el extractor avanza sobre el eje Z en un movimiento de aproximación al rack, es decir cuando el mecanismo telescópico se activa y está completamente extendido:

Figura 32: Mecanismo telescópico totalmente extendido

En general el desempeño del diseño mecánico ha sido bastante bueno en su relación costo – beneficio, recordando que la selección de materiales estuvo basada principalmente en el presupuesto del proyecto.

11.4 Ingeniería de Sistemas

Hay mucho que decir sobre este apartado, ya que sentar las bases del sistema es el objetivo central de este trabajo.

Primero mencionar que el funcionamiento planteado como primera iteración ha sido un éxito. El sistema cumple en la mayoría de aspectos con las metas iniciales del proyecto.

Empezare por hablar del comportamiento observado contra el planeado en operación:

Cada módulo hace la secuencia debida en el momento debido, hacen su trabajo y esperan de modo correcto a que los de más concluyan el suyo, si bien se han observado puntos que requieren atención, por ejemplo:

La secuencia del robot a home: La secuencia a home del robot está pensada de modo que cada módulo la ejecute simultáneamente, es decir que los módulos Y, Z y X inicien al mismo tiempo el proceso de establecer su punto de inicio “home”, pero se observó que cuando el mecanismo telescópico esta extendido completamente y a

la vez se inicia movimiento de ascenso o descenso por el eje Y, se crean movimientos indeseados que hacen inestable el sistema, aunque la idea es que el mecanismo extractor jamás este extendido al iniciar o finalizar la operación del sistema, es un comportamiento que actualmente está permitido, por lo que se cambiara la secuencia de inicio para asegurar que el mecanismo telescópico se encuentra retraído al momento de realizar movimiento en los demás ejes.

En cuanto a las comunicaciones del robot, una vez resueltos los problemas de ruido, la información transmitida por el bus era la esperada y aunque cada módulo recibía todos los mensajes del bus solo atendía como se esperaba los mensajes que iban dirigidos a él, aquí hare una breve recapitulación de lo pretendido para pasar a las reflexiones finales acerca de este punto:

El robot debe recibir la información inalámbricamente mediante Wifi, seguido de esto esa información es pasada al módulo principal cuyas funciones primordiales son ser el moderador de los módulos de movimiento asi como transmitir la información de las Peticiones a estos, el cual denomine “módulo principal”.

Aunque complete la librería de comunicación Wifi con un servidor MQTT para la recepción y publicación de mensajes, esta no se implementó debido a que la interfaz gráfica de usuario no pudo ser completada, ya que requería el desarrollo de bases de datos, seguridad y sobre todo la parte creativa del proyecto fue un reto difícil de superar, ya que la mayoría del tiempo que había presupuestado tuve que usarlo para hacer el diseño mecánico del prototipo, la cual requirió su tiempo y fue un esfuerzo creativo importante, también completar la construcción del prototipo diseñado, asi como escribir el software para cada módulo y desarrollar la electrónica del proyecto, sin mencionar todo el retrabajo necesario en cada una de estas áreas, asi como la escritura de este documento, lo cual ocasiono que el desarrollo total del proyecto tomase mucho más tiempo que el planeado originalmente.

Además del retrabajo y complejidad inherente al desarrollo de cada parte fundamental del robot se añaden los extensos tiempos de entrega para motores, sensores y demás componentes provenientes de China.

Como se probó la funcionalidad del prototipo actualmente fue del siguiente modo:

Manualmente programe en los buffers destinados a recibir esta información mediante Wifi los datos de algunas Peticiones, entonces el módulo principal con esa información procedía al análisis y a armar el paquete de datos destinado a cada módulo encargado del movimiento coordenado, el módulo principal también monitorea el estado de estos mensajes, es decir si fueron enviados o no.

Si alguno de los mensajes no fue enviado el módulo lo sigue retransmitiendo y la ejecución del prototipo es pausada indefinidamente con motivos funcionales y

también de seguridad, ya que es un conjunto de nada serviría que el eje Y si se mueva a donde se tiene que mover si el eje X no sabe a donde tiene que ir por no haber recibido el mensaje correctamente.

El comportamiento en todos los test fue el esperado, cuando intencionalmente llenaba el mensaje con datos incorrectos el módulo que recibía el mensaje no realizaba ninguna acción y esperaba que un mensaje con información correcta llegara al su destino.

En este apartado también ocurrió un error que me costó un poco de trabajo identificar, el cual fue el siguiente:

Si el módulo principal enviaba por alguna razón un mensaje que ya había sido recibido por alguno de los módulos X, Y, Z, se rompía la secuencia de funcionamiento.

Esto fue un error en el diseño del software que no considero tal escenario, lo cual fue corregido.

Derivado de las observaciones hechas durante las horas de prueba realizadas es que surgieron muchos de los puntos que se han planeado como trabajo futuro los cuales son medulares para un producto que va a estar en el mundo real funcionando de modo seguro.

Por ejemplo, es imperativo tener un registro de las causas de mal función que puedan llegar a presentarse, de modo que puedan corregirse, lo que lleva a otro punto importante que es poder actualizar el software del producto a la brevedad de manera remota.

También es importante añadir diversos mecanismos de ciberseguridad y protección contra malware.

Otra carencia importante en la actual implementación es que, si se corta la energía a mitad de la operación, el prototipo es incapaz de retomar el trabajo desde donde se quedó antes de la interrupción de energía, lo cual debe ser corregido creando un registro de operaciones para asegurar una operación segura y sin contratiempos en la mayoría de escenarios posibles.

12 CONCLUSIONES Y COMENTARIOS

Los alcances del proyecto fueron demasiado optimistas, cada parte del prototipo representaba un reto técnico bastante peculiar y requería especialización en diferentes campos de la ingeniería.

El propósito original era crear todo el entorno requerido para tener un sistema automatizado de biblioteca, Conveyor, la interfaz HMI, asi como los racks os sistemas de almacenaje con las unidades.

Como mencione por retrasos en muchos aspectos del proyecto, lo tuve que acotar únicamente al desarrollo del robot ASRS, el robot actual debo decir que me deja bastante satisfecho, es capaz de ejecutar las órdenes de trabajo con precisión, la secuencia de movimiento y el funcionamiento es el que se estableció en este documento y ha cumplido con el propósito de esta tesis que es crear un punto de partida para desarrollar uno de estos sistemas, esta primera iteración además ha revelado mejoras necesarias para crear un sistema real, las cuales no pueden ser ignoradas en el desarrollo de un sistema de tal complejidad como lo son los ASRS.

Realizar este trabajo revelo muchos aspectos técnicos que no había considerado cuando inicie en este proyecto, prácticamente cada aspecto del sistema puede y debe ser mejorado sustancialmente.

De las cosas que más disfrute hacer fue el software del robot, de hecho actualmente tengo un trabajo que disfruto demasiado como ingeniero de firmware para uno de los principales OEM del mundo en cuanto a Infoentretenimiento, BMS “Baterry Management Systems” por sus siglas en inglés, además de ADAS “Advanced Driver Assistance Systems”, lo que me ha brindado la maravillosa oportunidad de conocer cómo es que el software embebido es escrito de manera profesional en una compañía de talla mundial acorde a normas y estándares internacionales, lo cual, me permite afirmar sin duda que el software escrito para el robot podría escalarse sin problemas, claro que es la primer versión y requiere mucho retrabajo en muchas áreas, como lo son:

- Integrar mecanismos de ciberseguridad
- Implementar un bootloader que permita restaurar a una version segura el software asi como instalar actualizaciones remotas
- Incluir un manejador de errores, crear los códigos de falla del sistema
- Mejorar el sistema de monitoreo del hardware de cada módulo

Independientemente de la solución a utilizar, para crear la automatizacion de la biblioteca los puntos anteriores deben ser considerados necesarios para la implementacion del software.

Ha sido una tarea titánica y muy enriquecedora desarrollar cada parte del proyecto. Lo que menos disfruté, sin duda alguna y más problema me dio, fue el diseño electrónico, aquí sin duda recomendaría integrar al proyecto a un ingeniero experto en el tema, el cual les

ahorrara muchos dolores de cabeza y tiempo desperdiciando revisando cosas que están bien implementadas.

Si bien creo que el protocolo de comunicación establecido en esta tesis es suficiente, podría ser mejorado y extendido, actualmente hago uso de RS485, un solo maestro coordina la comunicación con los demás módulos y si requiere información la solicita de cada módulo debido a la restricción que establece el protocolo de un solo maestro, una posible actualización podría ser utilizar el protocolo CAN que cuenta con arbitración del bus de comunicaciones por hardware, de este modo los módulos podrían estar publicando el estado en el que se encuentran por sí mismos sin necesidad de que el maestro este preguntándoles periódicamente como es el caso actual.

Otro punto que podría tenerse en cuenta es el uso de procesamiento multicore, actualmente los microcontroladores usados para controlar cada módulo no poseen tal característica, si bien la complejidad y el costo del desarrollo de software para una plataforma multicore aumenta considerablemente también lo hacen los beneficios.

Todo el robot podría ser operado por un solo sistema multicore, aunque considero que es mejor mantener el uso de múltiples microcontroladores para controlar y monitorear el hardware en cada eje coordenado.

Aunque como mencione muchas características deben ser integradas por lo que podría incluso ser necesario optar por soluciones multicore para controlar y monitorear el hardware con las funcionalidades extendidas requeridas para cada módulo de control de cada eje coordenado.

Una característica fundamental que tiene que ser integrada es el uso de un sistema operativo en tiempo real (RTOS por sus siglas en ingles), con el cual sin importar el número de núcleos del hardware utilizado, cada módulo podría trabajar simulando “paralelismo” en las operaciones, aunque la plataforma seleccionada no sea multicore, lo cual es necesario para ejecutar tareas de monitoreo, gestión de errores y ejecutar secuencias de movimiento de forma casi paralela con hardware sin capacidades multicore o paralelas con hardware con capacidades multicore, los beneficios son muchísimos, por lo que se vuelve una necesidad el uso de un RTOS.

Actualmente el robot no cuenta con un sistema operativo en tiempo real, por lo que en varios puntos de la secuencia de funcionamiento podría quedarse “atorado” en la ejecución del programa si es que algún error desconocido ocurre y no hay funciones de monitoreo de errores, ni mucho menos un manejador de errores, para lo cual nuevamente hago énfasis en que es una necesidad operativa no negociable implementar un sistema operativo en tiempo real y un manejador de errores en el sistema real.

En cuanto al diseño mecánico quizá no haya mucho de valor que decir ya que el diseño del prototipo para nada podría ser considerado como una base para desarrollar el diseño

mecánico final de un sistema real de transelevador de columna y mucho menos es esa la intención de este trabajo. Aunque claro que puede aportar ideas al diseño final y a continuación brindare algunas conclusiones de mi diseño:

- El mayor esfuerzo de diseño debe darse en los soportes de la columna, los cuales la unen en este caso al tren inferior responsable de la traslación longitudinal en el eje coordenado X.
- Es importante considerar un modelo matemático de aceleración - desaceleración para mantener la carga dinámica dentro de rangos apropiados, el cual deberá ser implementado por software para que cuando arranque o frene el sistema lo haga de un modo más optimo.
- Si la distancia entre ejes para la cadena se alarga será necesario colocar guías para evitar que las cadenas se cuelguen por acción de la gravedad, asi mismo hay que mejorar el mecanismo tensor de estas.

13 TRABAJO FUTURO

- Como trabajo futuro está pendiente la implementación de un sistema operativo en tiempo real para el software del robot ASRS.

- Implementar un bootloader, de este modo el prototipo podría recibir actualizaciones remotas, además de contener firmware de restauración a una version estable y segura como método de seguridad.

- Analizar el código para asegurar que este escrito acorde a estándares de seguridad y buenas prácticas como MISRA.

- Implementar drivers de protección de escritura a regiones de memoria específicas.

- Implementar software para monitorear el estado de los componentes principales: Temperatura de los motores, corriente consumida por el sistema, voltaje de suministro, carga dinámica en los componentes mecánicos.

- Implementar funciones para la gestión de errores y códigos de falla.

- Implementar la comunicación Wifi con un servidor MQTT para el envío y recepción de Peticiones e información.

- Mejorar los drivers de intercambio de comunicación para incluir verificación CRC (Cyclic Redundancy Check).

- Implementar el registro de operaciones interno del robot del robot, el cual será almacenado en alguna memoria EEPROM o SD.

- Mejorar el diseño electrónico de las tarjetas de cada uno de los módulos asi como mandarlas a fabricar de modo profesional.

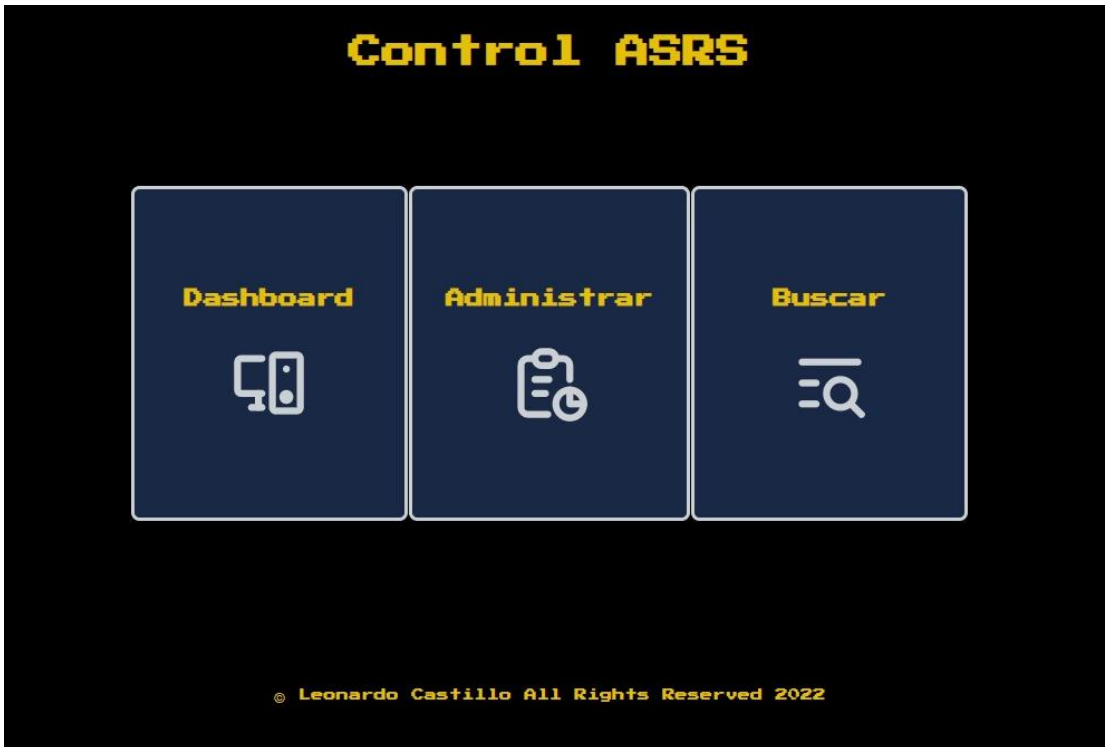
- Reemplazar todas las piezas fabricadas en impresión 3D por fundición de aluminio o aluminio maquinado.

- Mejorar el mecanismo de extracción

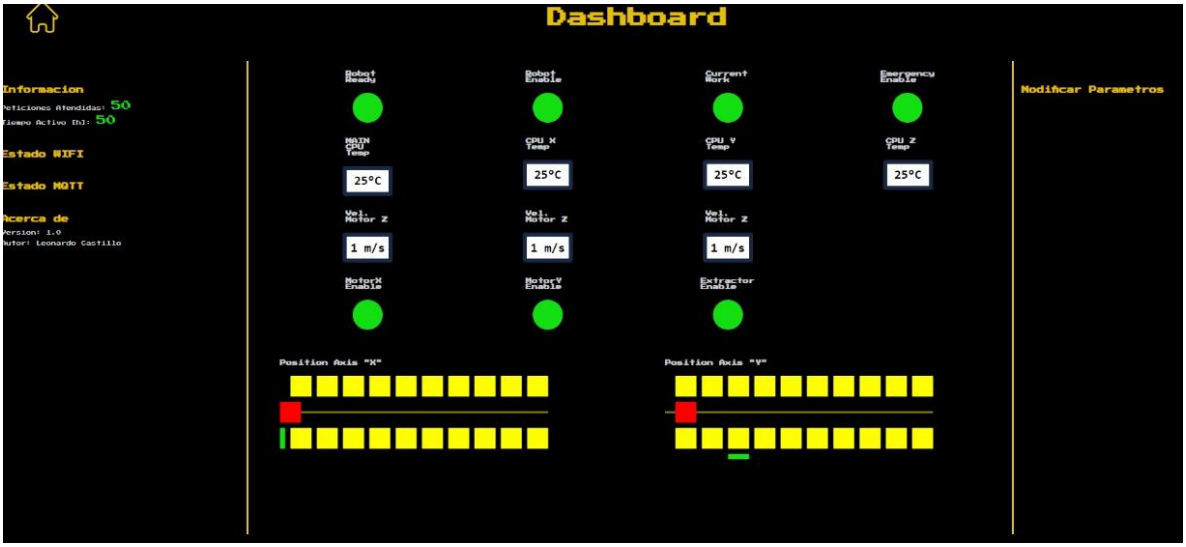
- Crear un algoritmo de aceleración- frenado para el prototipo, para que la carga dinámica inicial sea menor en los soportes de la columna.

- Implementar los Conveyor de trasporte, y en si el entorno faltante para recrear el entorno de un ASRS de mejor manera.

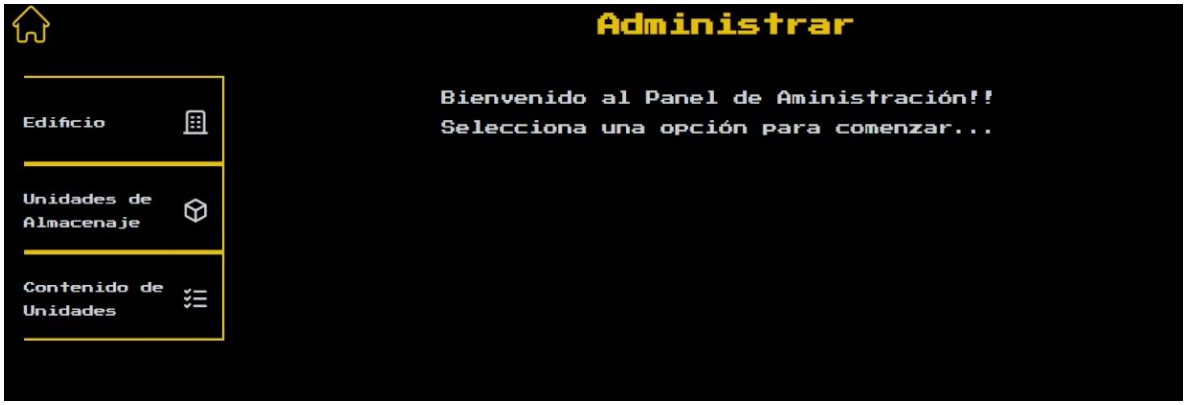
- Terminar la interfaz gráfica de usuario, la cual por sí misma tiene un buen nivel de desafío tanto técnico como creativo. A continuación, se muestra el estado actual del dashboard principal:



2. Ventana Inicial HMI



***3**. Dashboard HMI*



***4**. Panel Administración HMI*

5. Vista Inventario HMI

14 ANEXOS

14.1 Anexo A: Detalle de la implementacion del patrón de suscripción en lenguaje C.

```
 *      Author: Leo Castillo
 */
#ifndef INTMANAGER_H_
#define INTMANAGER_H_

#include <stdint.h>
#include <stdio.h>

#define SET_BIT_(n, BIT_F)        (n |= BIT_F)
#define UNSET_BIT(n, BIT_F)       (n &= ~BIT_F)
#define IS_SET(n, BIT_F)          (n &  BIT_F)
#define TOGGLE_BIT(n,BIT_F)       (n ^= BIT_F)    //XOR OPERATOR
#define COMPLEMENT(n)             (n = ~n)        //One´s Complement Operator

//Max number of clients that int server can holds
#define MAX_CLIENTS     10


/*Enum that hold all interrupt events that will be generated*/
typedef enum {

    DATA_RECEIVED = (1<<0),
    LIMIT_SENSOR1 = (1<<1),
    LIMIT_SENSOR2 = (1<<2),
    LIDAR_DATA    = (1<<3)

}tInterruptEventHandler;


//Callback funtion for notification
typedef void(*callback_fn)(void* pObject, tInterruptEventHandler Event);


typedef struct notify_handler {

    uint32_t Events;               //variable that hold the events at which the client is subscribed
    void *pObject;                 //Pointer to client object
    callback_fn fn_handler;        //Pointer to Callback function to execute

}notify_handler_t;


typedef struct interrupt_server {

    uint32_t n_client;
    notify_handler_t clients[MAX_CLIENTS];

}interrupt_server_t;

//Instance of server
interrupt_server_t InterruptServer;
```

```
void serverNotifyClients (interrupt_server_t * server, tInterruptEventHandler Event){

    for(int i=0; i<MAX_CLIENTS;i++){
        //verify that the client has registered its callback function
        if(server->clients[i].pObject !=NULL && server->clients[i].fn_handler != NULL){
            //verify that the client is subscribed to the event
            if(SET_BIT_((server->clients[i].Events), Event)){
                server->clients[i].fn_handler(server->clients[i].pObject, Event);
            }
        }
    }

}
```

```
void interruptManager (tInterruptEventHandler Event) {

    switch (Event) {

        case  DATA_RECEIVED:
            serverNotifyClients(&InterruptServer, DATA_RECEIVED);
            break;

        case LIMIT_SENSOR1:
            serverNotifyClients(&InterruptServer, LIMIT_SENSOR1);
            break;

        case LIMIT_SENSOR2:
            serverNotifyClients(&InterruptServer, LIMIT_SENSOR2);
            break;

        case LIDAR_DATA:
            serverNotifyClients(&InterruptServer, LIDAR_DATA);
            break;
```

```
void interruptServer_Init(interrupt_server_t * const server){

    for(int i =0; i<MAX_CLIENTS; i++){
        server->clients[i].fn_handler = NULL;
        server->clients[i].pObject = NULL;
    }

}
```

```
void modulClientSubscribe ( modulState_t * client, interrupt_server_t * server,tInterruptEventHandler Event){

    for(int i=0;i<MAX_CLIENTS;i++){

        //Check if client is already subscribed
        if(server->clients[i].pObject == client){
            //if the client is not subscribe to the event that is passed to the function yet
            if(IS_SET((server->clients[i].Events),Event)){
                //The client is already subscribed
                return;
            }else {
                //Subscribe the client to the event
                SET_BIT_((server->clients[i].Events), Event);
            }
            return;
        }

        //If client is not subscribed at any event yet.
        if(server->clients[i].pObject == NULL){
            server->n_client++;
            server->clients[i].pObject = client;
            server->clients[i].fn_handler = client->fn_handler;
            SET_BIT_((server->clients[i].Events), Event);

            //Remove duplicate
            for(int j = i+1; j<MAX_CLIENTS;j++){
                if(server->clients[j].pObject == client){
                    server->clients[j].pObject = NULL;
                    server->clients[j].fn_handler = NULL;
                    server->clients[j].Events = 0;
                }
            }
        return;
        }
    }
}
```

```
void modulClientIntHandler (void* pObject, tInterruptEventHandler Event) {

    if (Event == DATA_RECEIVED){
        ((modulState_t *)pObject)->ModulFlags.dataReceived = true;
    }

}
```

```
void modulInitialize (modulState_t * pt){


    pt->ModulFlags.modulFree = true;
    pt->ModulFlags.mainModul_Ready = false;
    pt->ModulFlags.peticionPendiente = false;
    pt->fn_handler = modulClientIntHandler;

}
```

```
typedef struct{

    ModulFlags_t  ModulFlags;
    struct variablesState  _stateModul;
    callback_fn fn_handler;

} modulState_t;

modulState_t Modul2State;
```

15 Referencias

[1] [En línea]. Available: https://sps.honeywell.com/mx/es/products/automation/solutions-by-technology/asrs-systems.

[2] S. L. Y. Z. S. C. G. S. C. Jackrit Suthakorn, «An Enhanced Robotic Library System for an Off-Site Shelving Facility,» de *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference onVolume: 4*, 2002.

[3] S. D. B. A. Tarkesh.S.Pujari, «Design of Intelligent and Robotic Library System,» de *2017 2nd IEEE International Conference On Recent Trends in Electronics Information & Communication Technology (RTEICT)*, 2017.

[4] H. D. Shukla, «Development of an RFID-Based Semi-autonomous Robotic Library Management,» de *2020 4th International Conference on Automation, Control and Robots*, 2020.

[5] F. Maranesi, «A desicion-support system for the design and managment of warehouse systems,» *Computers in industry,* vol. 65, pp. 175-186, 2014.

[6] M.Vasili, «Automated Storage and Retrieval Systems: A Review on Travel Time Models and Control Policies.,» *Warehousing in the Global Supply Chain,* pp. 159-209, 2012.

[7] W.Zhigang, «Research on The Framework of Library Management System Based on Internet of Things,» de *13th International Conference on Measuring Technology and Mechatronics Automation*, Beihai,China, 2021.

[8] S. Kummar, «Conceptual Design of a Wi-Fi and GPS Based Robotic Library Using an Intelligent System,» *International Journal of Computer and Information Engineering,* vol. 9, nº 12, 2015.

[9] Y. Angal, «Automation in Library Management Using LabView,» de *International Conference on Computing Communication Control and Automation(ICCUBEA)*, Pune, India, 2016.

[10] W. Martono, «RFID-Based Intelligent Books Shelving System,» de *RFID Eurasia*, Istanbul, Turkey, 2007.

[11] V. Borja Ramírez y A. C. Ramírez Reivich, Innovación de Producto, Premio Nacional de Tecnología, 2006.

[12] V. J. Gonzáles Villela, *Metodología en el Desarrollo de productos.*

[13] Systems Engineering Handbook, NASA, 2007.

[14] P. S. Sajja, Essence of Systems Analysis and Design, Springer, 2017.

[15] K. T. Ulrich y S. D. Eppinger, Diseño y desarrollo de productos, Mc Graw Hill, 2009.

[16] Benewake, «www.benewake.com,» 22 febrero 2022. [En línea]. Available: www.benewake.com.

[17] R. A. Eric Bauer, Beyond Redundancy: How Geographic Redundancy Can Improve Service Availability and Reliability of Computer-Based Systems, IEEE, 2012.

[18] B. P. Douglass, Design Patterns for Embedded Systems in C, Elseiver, 2011.

[19] Desconocido, «mecalux,» 2022. [En línea]. Available: https://www.mecalux.com.co/almacenes-automatizados/almacenes-automaticos-para-cajas/transelevadores-cajas.

[20] Desconocido, «bionichive,» [En línea]. Available: https://www.bionichive.com/. [Último acceso: 2022].