# Construcción de un Sistema de Gestión de Stock completamente automatizado basado en la plataforma Arduino y protocolos IoT/Web

Luciana Florencia Aranda, Víctor Enrique Lamberto Jofré, Gonzalo Pichetti, Agustín Riva, Luciano Tonlorenzi

Universidad Nacional de La Matanza

aranda.luciana.f@gmail.com, enquibe@gmail.com, gpichetti@gmail.com, aagusriva@gmail.com, luciano.tonlorenzi@gmail.com

## Resumen

*En el presente documento se explicará la construcción y diseño de un Sistema de Gestión de Stock a través de un montacargas que se trasladará de forma autónoma dentro del depósito cuando necesite cumplir con las órdenes que recibirá sobre la gestión del inventario mediante un sitio web con el cual se comunicará por un módulo WiFi.*

*En este sitio web se podrá configurar la frecuencia con la que se llevará a cabo el control, consultar sobre el stock actual y aquellas anomalías que pudieran ocurrir en cuanto al mismo. También se podrán realizar solicitudes de ingreso y egreso de cajas de productos.*

*Para controlar el stock, el montacargas tendrá incorporado a su estructura un lector de código de barras para identificar los productos que debe contener la caja, así como características propias del mismo y celdas de carga que informarán si el peso de la caja se corresponde con las unidades que debe contener y, en caso de encontrar alguna anomalía, la misma será registrada e informada.*

**Palabras Clave**

Montacargas, Arduino, Motor Paso a Paso, Lector de Código de Barras, Seguidor de Líneas, CNY70, Control de Stock, MQTT, WiFi, Servidor, Sistema.

## Introducción

A partir de las exigencias en cuanto a eficiencia y productividad necesarias para ofrecer un servicio rápido, de calidad, confiable y con el menor costo posible, tener una correcta gestión del inventario tiene un papel primordial para lograrlo.

Llevar a cabo el control de stock forma parte de la logística de una empresa y consiste en organizar, planificar y controlar el conjunto de productos para evitar deterioros, robos, hurtos o cualquier otra incidencia que signifique la pérdida de ganancias.

Gestionar el inventario de forma manual podría parecer sencillo y da la sensación de mayor control al realizar el conteo y visualizar físicamente los productos. Sin embargo, dicha tarea debe realizarse de forma meticulosa y continua para lograr eficiencia, lo cual resulta ser una actividad tediosa y podría acarrear errores humanos.

Por lo mencionado anteriormente, incluir tecnología en estas actividades implementando un sistema que automatice las tareas permitirá disminuir errores, ahorrar tiempo y agilizar los procesos dentro de la empresa logrando realizar una mejor planificación y rentabilidad.

## Funcionalidades

El Sistema de Gestión de Stock propuesto se encarga de realizar las tareas necesarias para administrar el stock de manera correcta siendo llevado a cabo por un robot autónomo permitiendo mantener un inventario actualizado en tiempo real a través de 3 diferentes funcionalidades:

1. **Control periódico de stock**: El robot realizará una inspección completa de todo el depósito de forma periódica según la frecuencia indicada por el administrador.
2. **Ingreso de Stock**: A través de diferentes solicitudes gestionadas por un sitio web, el robot ingresará y registrará las nuevas unidades de producto que se deseen almacenar.
3. **Egreso de Stock**: Se podrá realizar también el proceso inverso al ingreso, es decir, el robot se encargará de efectuar la salida correcta de las cajas almacenadas en el depósito según los pedidos del administrador.

## Estructura

La misma fue diseñada teniendo en cuenta los siguientes conceptos fundamentales para, tanto el armado, como su durabilidad:

1. **Ligereza:** Para evitar un consumo excesivo de las baterías y de la potencia emitida por los motores encargados de la movilidad, la misma se realizó a base de aluminio.
2. **Robustez:** A su vez, la estructura debe poseer una gran densidad y firmeza para asegurar la durabilidad del robot.
3. **Maniobrabilidad:** Este concepto se encuentra muy relacionado con la *Ligereza*, ya que la estructura debe ser capaz de tener la facilidad de ser estable y manejable en algunos momentos de inercia para poder contrarrestarlos.

## Componentes Hardware

Los artefactos utilizados en la construcción del robot son:

- Microcontrolador Arduino Mega 2560
- Driver Puente H L298N
- Sensor infrarrojo CNY70
- Sensor ultrasónico HC-SR04
- Motor paso a paso
- Lector de código de barras con microcontrolador ESP8266 ATMEL
- Celda de carga e interfaz HX711
- Modulo WiFi ESP8266 NodeMCU v3
- Batería 12v 7ah

En la Figura 1 se puede observar cómo se encuentra distribuida la comunicación y la conexión entre los diferentes componentes del sistema.

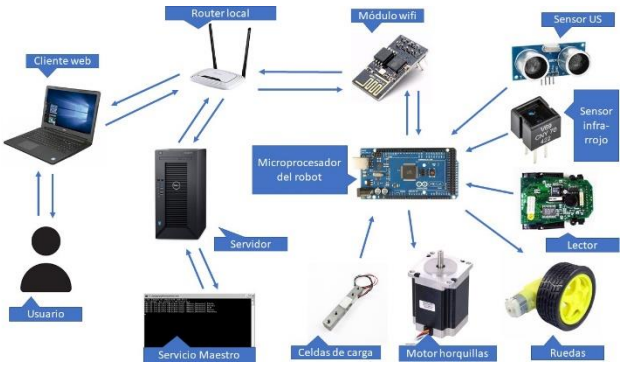


**Figura 1. Diagrama de Arquitectura**

Para el traslado cuenta con cuatro ruedas. Dos de ellas se encuentran en la parte trasera del robot y son las que poseen, cada una, los motores para el movimiento de este. Por otro lado, posee dos ruedas giratorias en la sección delantera para guiar en el recorrido y disminuir el peso de la carga.

Con respecto al procesamiento intrínseco del robot, se requiere de una placa de desarrollo con gran potencia para la puesta en marcha de los sensores y actuadores seleccionados.

El Arduino Mega 2560, que podemos ver en la Figura 2, está basado en el microcontrolador ATmega2560. Tiene 54 entradas/salidas digitales, 16 entradas analógicas, 4 UARTs (Transmisor-Receptor Asíncrono Universal), un cristal de 16Mhz, conexión USB, Jack para alimentación DC, conector ICSP (Programación serial en circuito), y un botón de reinicio. [1]

Se utilizó Arduino Mega por ser el más robusto de las placas, lo cual es apropiado para la complejidad del proyecto. Además, al ser de código abierto y flexible en cuanto a software y hardware, hay múltiples herramientas que facilitan su uso.


**Figura 2. Placa Arduino Mega 2560**

Aquellos motores que realizan la fuerza para el traslado utilizan una caja reductora con un sistema de engranajes para generar las RPM (Revoluciones Por Minuto) necesarias del movimiento. A su vez, se conectan al microprocesador a través del driver L298N (véase Figura 3) ya que, a través de dos puentes H, es el módulo más utilizado e indicado para manejar estos motores. Permite controlar el sentido y velocidad de giro de motores mediante señales TTL (lógica transistor a transistor) que se pueden obtener de microcontroladores y tarjetas de desarrollo como Arduino. El control del sentido de giro se realiza mediante dos pines para cada motor, la velocidad de giro se puede regular haciendo uso de modulación por ancho de pulso (PWM por sus siglas en inglés). [2]

**Figura 3. Driver Puente H L298N**

Para realizar el recorrido dentro del depósito utiliza sensores infrarrojos CNY70 conectados a la placa, que permiten lograr el seguimiento de líneas que se encontrarán en el suelo.

También cuenta con sensores ultrasónicos HC-SR04 que permitirán detectar objetos que obstaculicen el camino y evitar choques inesperados.

Otro de los componentes conectados a la placa es el módulo lector de código de barras con microcontrolador ESP8266 ATMEL que permitirá identificar unívocamente cada caja.

El movimiento de las horquillas se realizará mediante el mecanismo de tornillo sin fin con tuerca. Este evitará que se tenga que mantener magnetizado el motor mientras transportamos la carga. Las horquillas estarán unidas a la tuerca y el tornillo sin fin girará con un motor paso a paso conectado a través de un driver A4988 que posee la fuerza necesaria para sostener los pesos de ellas.

Para controlar el peso de las cajas necesario para el control del stock, se utilizan celdas de cargas conectadas cada una con una interfaz trasmisora Hx711 que amplifican las señales emitidas por las celdas de carga.

La comunicación vía WIFI con el servidor se realiza mediante la conexión de la placa con un módulo emisor/receptor denominado NodeMCU de la familia ESP8266.

Finalmente, el montacargas utiliza una batería de 12V 7ah recargable que otorga la alimentación para todos los componentes (Véase sección “Alimentación”).

## Traslado y recorrido autónomo

Para la conducción automática del robot, se utilizaron sensores infrarrojos CNY70 para lograr el seguimiento de líneas contrastadas en la superficie.

El CNY70 es un sensor óptico reflexivo que tiene una construcción compacta dónde el emisor de luz y el receptor se colocan en la misma dirección para detectar la presencia de un objeto utilizando la reflexión del infrarrojo sobre el objeto. La longitud de onda de trabajo es 950nm. Se puede ver el principio de funcionamiento del sensor en la Figura 4.

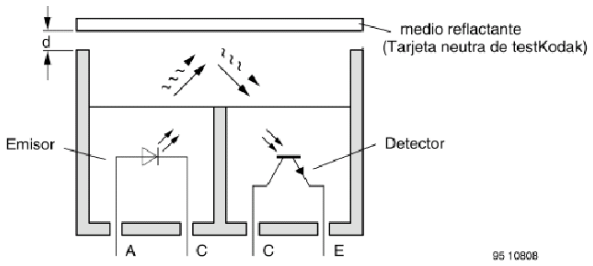


**Figura 4. Principio de funcionamiento del sensor.**

El principio en el que se basa el funcionamiento del sensor CNY70 es en la emisión de un haz de luz infrarroja por medio del diodo emisor. Dicho haz de luz se refleja sobre una superficie llegando así a la base del fototransistor. De esta manera, a medida que la superficie reflejante sea más clara, mayor corriente se producirá a través del fototransistor y así se obtendrá mayor voltaje a la salida y, mientras más oscura sea, menor será la intensidad de rayo infrarrojo reflejado, por lo que el voltaje de salida en el fototransistor será menor.

En este caso se utilizan 4 sensores infrarrojos ubicados de la siguiente forma: Dos de ellos se encuentran en la parte central inferior del montacargas y son los que permiten el seguimiento de las líneas que actuaran como guía para que el montacargas pueda circular en el depósito. Ambos sensores se encuentran juntos, uno al lado del otro y ocupan el mismo ancho que el de la línea a seguir. De esta forma, siempre buscamos que ambos detecten el color de la línea y, en caso de que uno no lo detecte, se aumentará la velocidad del motor de ese lado para girar al robot en sentido contrario y posicionarlo de forma correcta, tal como puede observarse en la Figura 5.

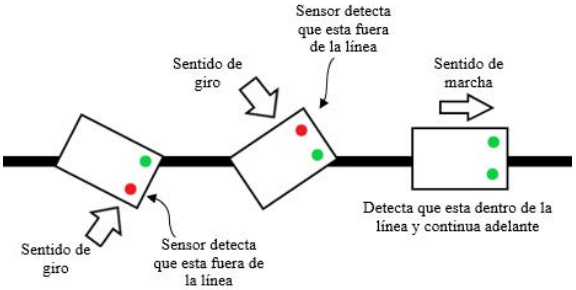


**Figura 5. Esquema de detección de línea**

Por otro lado, los dos sensores restantes, se encuentran más alejados del centro, uno del lado derecho y uno del lado izquierdo. En caso de que alguno de estos sensores detecte el color de la línea, significa que habrá un cruce y, de ser

indicado por el camino, el robot girara en el sentido correcto.

## Detección de obstáculos

Se utilizaron sensores ultrasónicos HC-SR04 como el que se muestra en la Figura 6, los cuales permiten detectar objetos que obstaculicen el camino y de esta forma evitar colisiones. Estos sensores tienen como ventaja una elevada precisión, lecturas estables y el funcionamiento no se ve afectado por material negro o luz solar. Además, no solo pueden detectar la presencia de un objeto, sino que también permite transmitir la distancia al mismo, lo cual es útil a la hora de evitar un choque ya que a partir de esa información se podrá recalcular el camino hacia el destino correspondiente.


**Figura 6. Sensor US HC-SR04**

## Levantamiento de las cajas

Para poder realizar el movimiento necesario para levantar las horquillas que a su vez levantaran los pallets de las cajas, se necesita tener la fuerza necesaria para sostener esos pesos, es por esta razón que se utilizó un motor paso a paso conectado a través de un driver A4988 como el que puede verse en la Figura 7. Una de las ventajas más importantes de un motor paso a paso es su capacidad para ser controlado con precisión y repetitividad en cuanto al posicionamiento en un sistema de lazo abierto, lo cual significa que no se necesita ninguna información de retroalimentación de posición logrando eliminar la necesidad de dispositivos de detección y regeneración como codificadores ópticos. [3][4]

**Figura 7. Motor Paso a Paso**

## Identificación de los productos

El montacargas debe obtener información acerca de los productos que contiene cada caja para así poder realizar una correcta gestión del stock. Es por esto que en su estructura cuenta con un módulo lector de código de barras QS2500 con un microcontrolador ESP8266 ATMEL como el que se puede apreciar en la Figura 8, que le permitirá al montacargas escanear el código que cada caja tendrá pegado en su parte frontal. De esta manera, una vez obtenido el código, se realizarán las consultas correspondientes a la base de datos para obtener las características de los productos que están en la caja, así como también su ubicación dentro del depósito.

**Figura 8. Módulo lector de código de barras QS2500**

## Cálculo de stock

La función principal del montacargas corresponde al control de stock mediante el pesaje de las cajas que contienen los productos. Para realizar la medición se utilizaron dos celdas de cargas ubicadas una en cada una de las horquillas que levantarán los pallets. Las mismas estarán conectadas cada una a una interfaz trasmisora Hx711 que permitirá amplificar las señales emitidas por las celdas de carga (Véase Figura 9).

La utilización de las celdas de carga se debe a que permiten transformar una presión mecánica en una señal eléctrica. A su vez, dicha señal eléctrica analógica es tomada por el HX711 y convertida a digital mediante un ADC de 24 bits y enviada a nuestro microprocesador.

Para determinar cuál es el stock actual de la caja que se pesó, primero el microprocesador obtiene la información del producto que debe contener la caja a partir de la lectura del código de barras que la identifica, y luego se compara con el peso obtenido mediante las celdas de carga. El resultado de la comparación determinará si el stock es el correcto o si existe algún tipo de diferencia. En caso de cumplirse esto último, se actualizará el inventario.

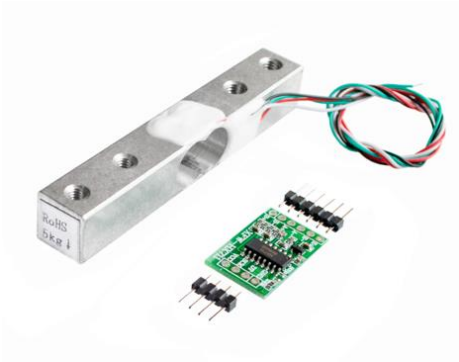

**Figura 9. Celda de carga junto con la interfaz HX711**

## Comunicación WiFi

El robot recibirá ordenes de ejecución según las solicitudes que vaya realizando el usuario. Por lo tanto, para poder captar estas solicitudes generadas desde un sitio web y procesadas y enviadas por el servidor, se utilizó como intermediario un módulo WiFi capaz de crear un cliente en la red y procesar dichas órdenes.

Como podemos observar en la Figura 10, este módulo, de la familia de los ESP8266, es la placa de desarrollo NodeMCU, que gracias a su gran capacidad de procesamiento y fácil instalación fue la indicada para este proyecto. [5]

**Figura 10. Módulo NodeMCU**

## Alimentación

Al manejarse de forma autónoma para reducir el personal dentro de los depósitos y así poder evitar robos, hurtos y deterioros, el montacargas deberá contar con la alimentación necesaria y con una gran durabilidad para poder funcionar correctamente. Además, los componentes hardware utilizados requieren entre 3 y 7.5v, a excepción del motor paso a paso que requiere una potencia de 12v.

Por lo mencionado anteriormente, para la alimentación general de todo el robot se decidió utilizar una batería de 12V 7Ah (Amperio-hora) recargable. El concepto de Amperio-hora indica la cantidad de carga eléctrica que pasa por los terminales cuando se proporciona una corriente eléctrica de 1 amperio durante 1 hora. Es decir que, al tener un consumo total de 500mA (miliamperios) entre todos los sensores y actuadores, la autonomía de la batería será de 14 horas.

Además, otra ventaja de dicha batería es que no posee un gran tamaño siguiendo con los estándares de tener una estructura liviana y maniobrable.

## Gestión de Comunicación

Además de la gran importancia que posee el robot autónomo, el corazón del proyecto es el servidor. Esto es así ya que este va a ser quien posea toda la lógica realizando los cálculos, comunicaciones y actualizaciones en la base de datos a través de diferentes algoritmos y procesos en segundo plano ejecutándose constantemente.

Decimos que el servidor es la parte lógica y funcional del sistema y que se encuentra desarrollada en tres partes bien definidas:

### a. Bróker Mosquitto MQTT

Para lograr la comunicación y el paso de mensajes entre todo el sistema lógico y funcional con el microprocesador del montacargas se utilizó el protocolo denominado MQTT definido por diferentes suscripciones y publicaciones de parte de cada uno.

MQTT es un protocolo de comunicación M2M (*machine-to-machine)* de tipo *message queue*. Está basado en la pila TCP/IP en donde cada conexión se mantiene abierta y se “reutiliza” en cada comunicación. Es una diferencia, por ejemplo, a una petición HTTP 1.0 donde cada transmisión se realiza a través de una nueva conexión. [6]

Como podemos observar en la Figura 11, el funcionamiento del protocolo MQTT se basa en un servicio de mensajería push con patrón publicador/suscriptor (pub-sub). Para filtrar los mensajes que son enviados a cada cliente, los mensajes se disponen en tópicos organizados jerárquicamente, en donde un cliente puede publicar un mensaje en un determinado tópico y, a su vez, otros clientes pueden suscribirse a este, generando que el bróker le haga llegar los mensajes suscritos. [7]

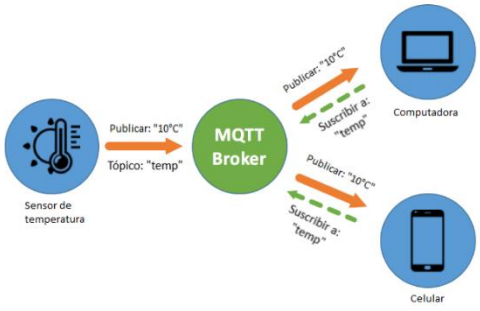


**Figura 11. Protocolo MQTT Publicador/Suscriptor**

Además, el protocolo MQTT dispone de distintas medidas de seguridad que podemos adoptar para proteger las comunicaciones. Esto incluye transporte SSL/TLS y autentificación por usuario y contraseña o mediante certificado.

Dicho protocolo fue elegido ya que se caracteriza por su sencillez y ligereza a la hora de la transmisión de mensajes, algo fundamental en el proyecto. Además, posee un consumo de energía menor y requiere un ancho de banda mínimo, lo cual es importante en redes inalámbricas, o conexiones con posibles problemas de calidad siendo importante debido a que como será implementado dentro de depósitos es altamente probable que las conexiones inalámbricas no sean las óptimas. Por lo tanto, es una solución largamente testada y consolidada, que aporta robustez y fiabilidad.

Por último, MQTT dispone de medidas adicionales importantes, como la seguridad y calidad del servicio (QoS) entendido como la forma de gestionar la robustez del envío de mensajes al cliente ante fallos (por ejemplo, de conectividad). Por una cuestión de confiabilidad en el paso de mensajes, fue necesario utilizar el nivel 2 de QoS ya que se garantiza que cada mensaje se entrega al suscriptor, y por única vez.

Como se ha mencionado, MQTT es un protocolo de comunicación, es decir que no es un software ya implementado, sino que ha sido necesario investigar en el mercado los diferentes productos existentes. Un software que implementa y ofrece servidores/brókeres con este protocolo es Mosquitto.

Mosquitto es un servidor de mensajes de código abierto (con licencia EPL/EDL) que implementa las versiones 3.1 y 3.1.1 del protocolo MQTT. La elección de este software fue debido a su ligereza lo que nos permite, fácilmente, emplearlo en gran número de ambientes, incluso si éstos son de pocos recursos.[8]

## b. Gestión de Rutas

El bróker posee un servicio programado en Python, gracias a su sencillez, que se encuentra ejecutándose indefinidamente realizando diferentes funcionalidades a través de distintos algoritmos. Estos últimos se pondrán en marcha según los mensajes que reciba tanto del cliente web como del robot, consultado y actualizando la base de datos según corresponda. Este servicio es tan importante como el robot debido a que es el que contiene el procesamiento total de la lógica a realizar en los diferentes casos a lo largo del tiempo.

También cabe mencionar acerca de un conjunto de operaciones sistemáticas que permiten calcular y hallar la ruta más corta para la movilidad del robot. Es decir, cuando este último debe dirigirse de una zona hacia otra, siempre se pondrá en marcha un algoritmo que obtendrá el camino mínimo de desplazamiento. La lógica utilizada en este caso fue la propuesta por Edsger Dijkstra en el año 1959.

El lenguaje en el que fue programado este servicio es Python. Su elección fue debido a que es uno de los lenguajes más simples, legibles y con gran cantidad de librerías existentes en el mundo de la programación. Al ser un lenguaje interpretado en vez de compilado, posee un régimen de legibilidad y transparencia que refieren a su filosofía. Además, es multiparadigmático el cual permite varios estilos de programación: orientada a objetos, imperativa y funcional. [9]

## c. Cliente Web

La tercer y última parte es el sitio web ya que es la cara visible hacia el mundo exterior y los posibles usuarios y compradores.

Entre las funcionalidades más destacadas que podemos encontrar en este cliente son:

1. **Gestión de solicitudes de ingreso y de egreso**: En esta sección, el administrador del depósito que haya adquirido el sistema podrá realizar solicitudes (tanto de entrada como de salida de productos) y enviarle dichas ordenes al robot para su puesta en marcha.
2. **Gestión del control de stock**: Aquí se podrá definir la periodicidad mencionada anteriormente (indicando los días de la semana y el horario de ejecución) para que el sistema y el robot realicen el control periódico en el momento indicado.
3. **Visualización de diferentes reportes y datos históricos**: Los administradores también contarán con una parte en donde podrán observar y también controlar todos los datos relevantes del depósito, como, por ejemplo: historial de cajas dentro, stock y ubicación actual de los productos, historial de las solicitudes realizadas y realizándose, etc.
4. **Situación actual del robot:** En esta sección se podrá visualizar la ubicación en tiempo real del robot dentro del depósito, como así también su estado de lo que vaya realizando.
5. **Gestiones administrativas**: Se podrán modificar diferentes aspectos y datos en cualquier momento.

Con respecto a la construcción, se pensó primordialmente en una página web completamente responsiva debido a los requisitos actuales de la sociedad. Por lo tanto, el sitio es completamente adaptable a cualquier tipo de pantalla, desde un teléfono celular hasta una pantalla tanto de televisión como de computadora.

A lo que refiere la programación, el cliente fue realizado con HTML5, CSS3 y JavaScript. Además, para la realización de las diferentes funcionalidades nombradas anteriormente, fue fundamental la utilización y creación de diferentes Web Services programados en PHP 7.1 y MySQL, en lo que respecta a la base de datos.

Por último, debido a que los servicios en segundo plano del servidor son los que poseen la lógica de realización, se necesitó encontrar una forma de comunicar este cliente web

con el bróker utilizando el protocolo WebSocket ya que proporciona un canal de comunicación bidireccional y full-dúplex sobre un único socket TCP, tal como se muestra en la Figura 12. [10]

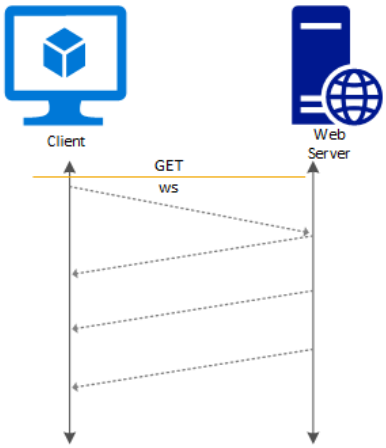


**Figura 12. Protocolo WebSocket**

## Conexión Local

Es sumamente importante que todas las partes se encuentren conectadas entre sí para realizar la comunicación eficiente que se necesita. Además, no solo se necesita que estén conectadas, sino que la red que lleve a cabo esto sea completamente fiable y con un error de servicio casi nulo ya que podría poner en riesgo la realización de las diferentes solicitudes. Sin embargo, el sistema y el robot poseen la capacidad y la inteligencia para adaptarse y finalizar sus tareas en ejecución en el caso de una caída en la conexión de la red.

Frente a esta problemática, se pensó en la creación de una red completamente local sin salida a internet para asegurar la seguridad en la no interferencia ni la ralentización de otros dispositivos y/o personas. Es por ello que fue necesaria la instalación de un router en ámbito local con un diagrama de direcciones IP fijas para la correcta comunicación.

## Resultados

En la construcción de Sistema de Gestión de Stock completamente autónomo, son numerosas las diferentes pruebas y cambios de diseño que ocurren. Sin embargo, se puede afirmar que, con una buena distribución y delegación de las diferentes partes, se pueden obtener los resultados esperados por el equipo.

Antes de comenzar su elaboración, integración de las diferentes partes y puesta a punto, fue necesario comprobar el correcto funcionamiento de cada uno de los componentes hardware por separado para poder obtener los rendimientos esperados.

En primer lugar, se realizaron las pruebas de los sensores infrarrojos que generan el seguimiento de la línea para la movilidad del robot. La necesidad era controlar la conexión y que estos emitan los valores deseados (entre 50 y 200 si la línea es oscura y entre 200 y 700 si la línea es clara) para corroborar que no se hayan deteriorado. Los resultados obtenidos arrojaron un valor de 120 para el primero caso y de 350 para el segundo.

En segundo lugar, se puso a prueba los sensores de ultrasonido y pudimos comprobar que el ángulo de funcionamiento y la sensibilidad de este es de 15 grados y 3 mm, respectivamente.

En tercer lugar, se realizaron las pruebas del motor paso a paso para conocer el peso máximo a poder utilizar. Una vez realizada las conexiones, tanto al microprocesador como al driver controlador del mismo, comenzamos a poner diferentes pesos en las horquillas y observar su fuerza, pudiendo obtener que permite una capacidad de hasta 5kg.

Por último, se ha verificado la distancia máxima y los diferentes ángulos a los que puede posicionarse correctamente el lector de código de barras. Los resultados obtenidos arrojaron que el dispositivo posee una distancia máxima de 30 cm a 15 grados de inclinación y 25 cm a 45 grados.

En la Tabla 1 podemos observar una descripción de los resultados obtenidos frente a los resultados esperados para los diferentes sensores.

**Tabla 1. Resultados de los sensores**

| Sensor / Actuador | Resultado Esperado | Resultado Obtenido |
|---|---|---|
| Infrarrojo (Oscuridad) | Entre 50 - 200 | 120 |
| Infrarrojo (Claridad) | Entre 200 - 700 | 350 |
| Ultrasonido (Distancia) | Mín.: 0cm<br>Máx.: 500cm | Mín. 2cm<br>Máx.: 400cm |
| Motor paso a paso (Peso) | Máx.: 5kg | Máx.: 5kg |
| Lector de código de barras (Distancia) | Mín.: 2cm<br>Máx.: 50cm | Mín.:5cm<br>Máx. 1: 30cm (15°)<br>Máx. 2: 25cm (45°) |

## Discusión

Si bien los resultados obtenidos en las pruebas realizadas fueron positivos, aún resta por comprobar la adaptabilidad del robot a una superficie rugosa o resbaladiza en donde dificulte su desplazamiento por el depósito.

De todas formas, podemos decir que los motores que generan la potencia para el movimiento poseen un torque (o par motor) aceptable para una primera versión y, también, que el equipo de desarrollo se encuentra realizando las

investigaciones necesarias para sacar adelante dicha problemática.

## Conclusión

Construir un robot autónomo junto con un sistema que procese y ejecute cada orden (tanto en la interfaz del usuario como en las ejecuciones del dispositivo) no es nada sencillo. Es un proyecto que involucra una gran variedad de conocimientos técnicos y funcionales de diversas áreas como electrónica, física, lógica (para los diversos procesos de ejecución) y software. También se requiere una gran capacidad de investigación y adaptabilidad para ir resolviendo las diferentes problemáticas que se van presentando.

Sin embargo, las pruebas demuestran que esto es posible y viable realizar esto y su costo es significativamente menor al costo del deterioramiento, robos y hurtos de los productos que se encuentran en el depósito.

Además, es importante poner en juego los beneficios obtenidos al lograr esta automatización de stock. En primer lugar, le damos la posibilidad a que diferentes empleados que antes se encargaban de hacer los recuentos de forma manual, ahora pueden dedicarse a otras tareas. Por otro lado, el desorden generado por los movimientos del flujo de mercadería y los recuentos físicos que un operario realiza siendo una tarea tediosa, hacen que disminuya considerablemente la eficiencia de la empresa. Y, por ultimo y más importante, logramos evitar considerablemente los accidentes ocurridos dentro del depósito, salvaguardando la salud física y mental del empleado.

Finalmente, debido a estas características y a que es un proyecto de gran interés para las fabricas con grandes depósitos, se espera recibir diferentes contribuciones y aportes de sponsors que deseen sustentabilizar y mejorar el producto.

## Agradecimientos

Desde la concepción del equipo de trabajo hasta la finalización del producto, fue necesaria la ayuda y la comunicación de diferentes entidades y personas. Es por esto que el equipo de desarrollo desea agradecer a:

- Universidad Nacional de La Matanza.
- Ing. Claudio D'Amico, Coordinador de la carrera Ingeniería en Informática.
- Docentes y Tutores de la Catedra Proyecto Final de Carrera.
- Grupo de profesores de las materias asignadas a la carrera.
- Esteban Carnuccio, profesor de la materia Sistemas Operativos Avanzados.
- Familiares y amigos del grupo de desarrollo.

## Referencias

[1] Comunidad Arduino. "Arduino Mega 2560". http://arduino.cl/arduino-mega-2560/

[2] Naylamp. "Driver Puente H L298N". https://naylampmechatronics.com/drivers/11-driver-puente-h-l298n.html

[3] Frank Mecafenix. "Motor paso a paso ¿qué es y cómo funciona?". https://www.ingmecafenix.com/electricidad-industrial/motor-paso-a-paso/

[4] Javier Arnedo. "Driver A4988 + Nema 17". http://www.javierarnedo.com/arduino/driver-a4988-nema-17-ajuste-voltaje-referencia/

[5] Luis Llamas. "Placa de Desarrollo NodeMCU ESP8266". https://www.luisllamas.es/esp8266-nodemcu/

[6] Luis Llamas. "¿Qué es mqtt? Su importancia como protocolo iot". https://www.luisllamas.es/que-es-mqtt-su-importancia-como-protocolo-iot/

[7] MQTT. "Servers/Brokers". http://mqtt.org

[8] Eclipse Foundation. "Eclipse Mosquitto™. An open source MQTT broker". https://mosquitto.org/

[9] Python Org. "Python Documentation". https://www.python.org/doc/

[10] Microsoft Azure. "Overview of WebSocket support in Application Gateway". https://docs.microsoft.com/en-us/azure/application-gateway/application-gateway-websocket