DEPARTAMENTO DE CIENCIAS DE LA ENERGÍA Y MECÁNICA

CARRERA DE INGENIERÍA MECATRÓNICA

TRABAJO DE TITULACIÓN, PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN MECATRÓNICA

TEMA: DISEÑO E IMPLEMENTACIÓN DE UNA ARQUITECTURA IOT PARA ROBÓTICA COLABORATIVA

AUTORES: GONZÁLEZ BONIFAZ, DAVID EFRAÍN

VERDUGO CABRERA, ALEXANDRA DEL CARMEN

DIRECTOR: ING. ESCOBAR CARVAJAL, LUIS FERNANDO

SANGOLQUÍ - 2018

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE CIENCIAS DE LA ENERGÍA Y MECÁNICA

CARRERA DE INGENIERÍA EN MECATRÓNICA

CERTIFICACIÓN

Certifico que el trabajo de titulación, ***"DISEÑO E IMPLEMENTACIÓN DE UNA ARQUITECTURA IOT PARA ROBÓTICA COLABORATIVA."*** fue realizado por el señor ***González Bonifaz, David Efraín*** y la señorita ***Verdugo Cabrera, Alexandra del Carmen***, el mismo que ha sido revisado en su totalidad, analizado por la herramienta de verificación de similitud de contenido; por lo tanto cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, razón por la cual me permito acreditar y autorizar para que lo sustente públicamente.

Sangolquí, 03 de julio del 2018.

Firma:

Ing. Escobar Carvajal, Luis Fernando
Director
C.C.: 1002403200

DEPARTAMENTO DE CIENCIAS DE LA ENERGÍA Y MECÁNICA

CARRERA DE INGENIERÍA MECATRÓNICA

AUTORÍA DE RESPONSABILIDAD

Nosotros, ***González Bonifaz, David Efraín*** y ***Verdugo Cabrera, Alexandra del Carmen***, declaramos que el contenido, ideas y criterios del trabajo de titulación: ***"DISEÑO E IMPLEMENTACIÓN DE UNA ARQUITECTURA IOT PARA ROBÓTICA COLABORATIVA"*** es de nuestra autoría y responsabilidad, cumpliendo con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Consecuentemente el contenido de la investigación mencionada es veraz.

Sangolquí, 25 de junio del 2018.

Firman:

David Efraín González Bonifaz

C.C.: 1718538034

Alexandra del Carmen Verdugo Cabrera

C.C.: 0301964318

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE CIENCIAS DE LA ENERGÍA Y MECÁNICA

CARRERA DE INGENIERÍA MECATRÓNICA

AUTORIZACIÓN

Nosotros, ***González Bonifaz, David Efraín*** y ***Verdugo Cabrera, Alexandra del Carmen***, autorizamos a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación: ***"DISEÑO E IMPLEMENTACIÓN DE UNA ARQUITECTURA IOT PARA ROBÓTICA COLABORATIVA"*** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de mi responsabilidad.

Sangolquí, 25 de junio del 2018.

Firman:

David Efraín González Bonifaz

C.C.: 1718538034

Alexandra del Carmen Verdugo Cabrera

C.C.: 0301964318

DEDICATORIA

Al amor de mi vida, a mi madre y a la ciencia.

David Efraín González Bonifaz

DEDICATORIA

A los mejores padres del mundo Walter y Alexandra, que a pesar del sacrificio de separarnos durante tantos años físicamente, siempre sus oraciones y apoyo incondicional me hicieron sentir la persona más amada y bendecida del mundo.

A mis adorados abuelitos y tíos, que con su apoyo constante me dieron la fuerza necesaria para seguir adelante y nunca dejar de luchar por alcanzar mis metas.

A mis hermanitos Oswaldo y Sebastián y a mis primitos, que son una partecita fundamental en mi vida y con este logro espero poder convertirme en su ejemplo y demostrarles que los sueños se pueden hacer realidad.

A mi amado novio, porque de su mano pude recorrer este camino que parecía imposible, pero al llegar a mi vida lo hizo sencillamente maravilloso, siendo siempre mi fortaleza y dándome la energía necesaria para saber levantarme incluso cuando todo parecía perdido.

Alexandra del Carmen Verdugo Cabrera

AGRADECIMIENTO

A mi madre, que con su constante paciencia y sabiduría ha sabido guiarme en los sinuosos caminos de la vida. Mi anhelo sigue siendo crecer y llegar a tener un espíritu tan férreo y pacífico como el suyo.

A Naomi, mi novia, que dio un nuevo matiz a mi vida y me inspiró, cuidó y aconsejo en los momentos más oportunos.

A mis hermanos Diego y Santiago que me motivaron a ser un ejemplo y trazar las pautas para un camino lleno de alegrías y descubrimientos.

A mi padre que estuvo siempre que lo necesité y que, indudablemente, es una parte fundamental en mi vida.

A mi abuelita Carmen que me acogió cuando empecé el camino universitario y me acompañó en las madrugadas y las malas noches.

A mis tíos Patricio, Hernan y Cecilia. De quienes siempre hubo palabras de aliento en los momentos más oscuros y el apoyo incondicional para poder llegar hasta aquí.

A todos mis amigos, profesores y compañeros de la universidad que rompieron mis prejuicios y me acompañaron a ver el mundo con ojos científicos.

David Efraín González Bonifaz

AGRADECIMIENTO

A Diosito, por ser el autor de la profesional en la que me he convertido, por haber guiado mis pasos a lo largo de estos años y por nunca dejarme sola.

A mis padres, el más grande ejemplo que tengo de lo que la constancia y el estudio pueden lograr, unos profesionales que en base a su esfuerzo y sacrificio han sabido sacar adelante a su familia permitiendo que sus hijos nos convirtamos en los seres humanos que somos actualmente.

A mi abuelito Carlos porque este espacio se lo debo a él, quien me enseñó toda la vida que uno de los más grandes valores que debe tener el ser humano es la gratitud.

A mi familia entera que, a pesar de no estar junto a mí, siempre mi corazón estuvo con ellos y fue lo que me dio la fuerza necesaria para cumplir este reto que parecía imposible pero sus oraciones y bendiciones me hicieron salir adelante y cumplir mi sueño, para que nunca dejaran de sentirse orgullosos de mis logros.

A mi novio, mi más grande aliado en esta batalla, quien no solo estuvo pendiente de mí y me ayudó en cada paso que di, sino que también supo compartir conmigo el cariño de su familia, permitiéndome encontrar personas maravillosas que me han ofrecido su apoyo, aprecio y compañía incondicionales.

A mi hermano del corazón Gonza, por haberse convertido en el más grande amigo que jamás creí encontrar, sin permitir que me rinda nunca e incentivándome a convertirme cada día en una mejor versión de mí misma.

A mis amigos, personas con las que no solo compartí en las aulas de clase, más bien personas que se convirtieron en mi familia de corazón y con los que llegué a compartir grandes aventuras que se han convertido en maravillosos recuerdos y experiencias que tendré para toda la vida.

Alexandra del Carmen Verdugo Cabrera

ÍNDIDE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

La evolución del internet ha dado paso a la inclusión de objetos que hacen uso de los servicios de la red mundial sin necesidad de la intervención del ser humano, concepto que se engloba bajo el término de Internet de las Cosas (IoT por sus siglas en inglés). El crecimiento exponencial de los objetos conectados ha influido en zonas productivas clave del desarrollo tecnológico mundial, tales como Ciudades Inteligentes, Industria 4.0 y Robótica en la Nube. Sin embargo, debido a la falta de plataformas accesibles por parte de los estudiantes y docentes investigadores, el estudio de esta tecnología dentro de la Universidad de las Fuerzas Armadas – ESPE es mínimo. El trabajo detalla el diseño e implementación de una arquitectura IoT para robótica colaborativa. El diseñó se basó en la norma VDI 2206 para sistemas mecatrónicos. El proyecto está formado de tres componentes principales que son el sistema Multi-Robot, la arquitectura de comunicación y el planificador automático. Las pruebas se enfocaron en evaluar criterios IoT (Tecnología distribuida, Interacción de objetos conectados, Seguridad, Escalabilidad y Eficiencia Energética), el tiempo de respuesta, el costo computacional, el sistema Multi-Robot y el planificador automático. Los resultados fueron satisfactorios con lo que se concluyó que las prestaciones de la arquitectura implementada permiten la comunicación y escalabilidad del sistema Multi-Robot aplicado en tareas colaborativas.

Palabras claves

- **ARQUITECTURA IOT**
- **MQTT**
- **SISTEMA MULTI-ROBOT**
- **PLANIFICADOR AUTOMÁTICO**

ABSTRACT

The evolution of the internet has given way to the inclusion of objects that make use of the global network's services without the need for human intervention, a concept that is included under the term Internet of Things (IoT). The exponential growth of connected objects has influenced important production areas of global technological development, such as Smart Cities, Industry 4.0 and Cloud Robotics. However, due to the lack of accessible platforms by students and teacher researchers, the study of this technology within the University of the Armed Forces - ESPE is minimal. The work details the design and implementation of an IoT architecture for collaborative robotics. The design was based on the VDI 2206 standard for mechatronic systems. The project consists of three main components, the Multi-Robot system, the communication architecture and the automatic planner. The tests focused on evaluating IoT criteria (Distributed Technology, Interaction of connected objects, Security, Scalability and Energy Efficiency), the response time, the computational cost, the Multi-Robot system and the automatic planner. The results were satisfactory with what was concluded that the features of the implemented architecture allow the communication and scalability of the Multi-Robot system applied in collaborative works.

Keywords

- **IOT ARCHITECTURE**
- **MQTT**
- **MULTI-ROBOT SYSTEM**
- **AUTOMATIC PLANNER**

CAPÍTULO 1

GENERALIDADES

El capítulo detalla los antecedentes que dieron partida al proyecto mediante una corta revisión histórica de los conceptos principales que lo componen. Después se describe la justificación e importancia que motivaron a su desarrollo. Se presenta los componentes, funciones y objetivos del proyecto finalizando con una breve descripción de la estructura del documento.

1.1. Antecedentes

1.1.1. El Internet de las Cosas: Origen e Impacto

El internet marcó el nacimiento de una nueva era tecnológica, la cual se caracteriza por la capacidad de la información de viajar atreves del mundo en segundos. Desde su creación en 1991 según (Pavón, 2012), el World Wide Web [1] puso miles de bytes de información al alcance de cualquier persona con un computador, cifra que actualmente se aproxima a los 5 millones de Terabytes (Pulido Cañabate, 2016). El internet evolucionó hasta ser capaz de proveer una vía de comunicación global y ser fuente de acceso a datos de toda índole. Sin embargo, con el avance de los microcontroladores, microprocesadores y demás sistemas computacionales, los horizontes del internet se extendieron.

Hasta inicios del siglo XXI toda la información disponible en la red necesitó de un intermediario humano. Todo dato había sido tecleado, grabado o registrado por una persona para ser subido al internet. De acuerdo a (Madakam, Ramaswamy, & Tripathi, 2015) en 1999, Kevin Ashton se enfrentó a esta problemática proponiendo una red en la cual los sistemas informáticos

[1] El origen del internet se puede referir a varios años antes de la creación del Word Wide Web, pero este punto es clave en la globalización del internet.

interpretan el entorno haciendo uso de sensores que le proveen datos directamente. La idea de Ashton consistió en hacer partícipes del internet a los electrodomésticos, medicamentos, vehículos y demás cosas que rodean al ser humano. Lo cual permitiría que situaciones como fármacos caducos, escases de stock e inventarios desactualizados no se vuelvan a producir. Ashton acuna el término Internet de las Cosas (IoT por sus siglas en inglés) refiriéndose a la independencia de los objetivos en el uso del internet para mejorar sus prestaciones.

Cisco define el IoT como un punto en el tiempo entre el 2008 y 2009 en el que hubo más dispositivos que personas conectadas al internet (Evans, 2011). El IoT ha tenido un enorme impacto en las zonas productivas, logísticas, científicas y tecnológicas de la sociedad. Entre los campos prioritarios de aplicación del IoT se encuentra la industria 4.0 cuyos objetivos son la optimización de recursos, automatización de procesos, entornos seguros de trabajo, entre otros. Las ciudades inteligentes que buscan mejorar la calidad de vida de los ciudadanos mediante atención eficiente de emergencias, responsabilidad medioambiental, control del tráfico, etc. y la Robótica en la Nube que busca la explotación de los recursos en línea para globalizar los avances robóticos y dotar de mayores prestaciones a los robots.

Los estudios IoT realizados en el país según (Abasolo Aranda & Carrera Paz y Miño, 2014) están orientados a la implantación de arquitecturas basadas en plataformas comerciales, open hardware y conectividad IPv6. Así mismo (Peña Merizalde & Suquillo Chuquimarca, 2016) lo estudiaron para la implementación de un prototipo domótico que consta de tres sistemas importantes: seguridad, iluminación y climatización. Por otra parte, (Rodriguez & Geovanny, 2015) consideran la seguridad del modelo, donde se analizan las debilidades y amenazas que tiene esta tecnología.

1.1.2. El Internet de las Cosas y la Robótica: Robótica en la Nube

La Robótica en la Nube es un campo de la robótica que busca hacer uso de tecnologías en la nube como procesamiento, almacenamiento, comunicación, aprendizaje, etc., con el fin de mejorar las prestaciones de los robots (Oussama Khatib, 2008).

Al crear una nube para robots se busca que centros de datos, aprendizaje profundo, modelos de entorno, soporte de comunicación y otros desarrollos de robótica se puedan aprovechar por la comunidad global (Portilla & Guzmán, 2013). La Robótica en la Nube, al igual que similares campos tecnológicos emergentes, se enfrenta a retos que impiden su auge, entre ellos, arquitecturas IoT para robótica. El diseño e implementación de estas arquitecturas debe facilitar la interconexión de robots, personas y dispositivos bajo una misma red global.

A nivel internacional se han desarrollado investigaciones de este tema, por ejemplo, el proyecto de maestría propuesto por (Larrañaga Fuerte, 2016) en el cual crea una solución IoT integrando dispositivos National Instrument con software de IBM. Por otra parte (Marroquin, Gomez, & Paz, 2017) elaboraron un proyecto que consiste en un robot móvil de bajo costo para la exploración de lugares implementado con una arquitectura IoT que permite controlar al robot desde una Aplicación Web.

1.1.3. Arquitectura IoT para Robótica Colaborativa

La robótica colaborativa da un enfoque diferente al trabajo conjunto entre dos o más robots. El propósito principal es prescindir de una funcionalidad específica para cada uno y en su lugar poseer una programación global que se encargue de cumplir los objetivos para los que fueron diseñados, de esta forma se busca eliminar redundancias en su funcionamiento, optimización de recursos y un mayor nivel de automatización.

A nivel nacional es muy poca la información que se puede recabar. Sin embargo, existen estudios como el propuesto por (Campoverde, 2017) en la Universidad de las Fuerzas Armadas – ESPE, titulado "Desarrollo de dos robots para realizar trabajo cooperativo". En dicho proyecto los robots forman figuras geométricas básicas siendo la base para futuros trabajos relacionados. De igual manera en la misma Universidad se cuenta con el proyecto realizado por (Tabango, 2014), que utiliza dos robots humanoides que trabajan colectivamente para mover una pelota de un lado a otro.

1.2. Justificación e Importancia

El Internet de las Cosas ha sido la base de avances tecnológicos importantes en las últimas décadas. Su uso en la robótica, junto con otros hitos informáticos, forman los componentes esenciales de la Robótica en la Nube. Entre los retos que se deben superar para alcanzar su máximo potencial se encuentra el desarrollar plataformas IoT para robótica. Dichas plataformas deben responder a la problemática de unir a dispositivos, personas y robots en una sola red. En vista de la importancia que tiene el internet de las cosas en el desarrollo actual y en la robótica, es oportuno disponer de plataformas IoT, conceptuales o implementadas, en la Universidad de las Fuerzas Armadas – ESPE que faciliten el aprendizaje e investigación por parte de los miembros de la universidad.

En el proyecto se diseñó e implementó una Arquitectura IoT para robótica colaborativa que estará a disposición del Laboratorio de Mecatrónica y Sistemas Dinámicos de la universidad. La cual es fácilmente escalable y replicable, de modo que las pruebas, investigaciones y desarrollos por parte de los estudiantes y docentes-investigadores, se puedan ejecutar sobre esta plataforma.

1.3. Alcance

El proyecto consistió en el diseño e implementación de una arquitectura IoT para robótica colaborativa. Los componentes principales del proyecto son un Sistema Multi-Robot, una arquitectura IoT y un planificador automático. El sistema Multi-Robot tiene tres robots móviles con ruedas controlados por una tarjeta NI MyRIO (National Instruments). Los cuales tienen un sistema de navegación basado en odometría y navegación inercial que les permite ubicarse en un área de seis metros cuadrados. La arquitectura IoT se basa en el protocolo MQTT. El bróker que gestiona la comunicación y la interfaz de usuario son accesibles desde internet. El planificador automático se encarga de la canalización de datos provenientes de la interfaz de usuario hacia los robots y el control colaborativo centralizado. La aplicación para el sistema fue la clasificación de piezas en función de su color. La prioridad de organización de los colores es determinada por el usuario.

1.4. Objetivos

1.4.1. Objetivo General

Diseñar e implementar una arquitectura IoT para robótica colaborativa.

1.4.2. Objetivos Específicos

- Diseñar una arquitectura IoT capaz de recibir información de robots móviles, almacenarla, procesarla y determinar una acción adecuada.
- Basar el diseño de la arquitectura IoT en software libre y algoritmos escalables y compatibles con plataformas comerciales de uso extendido.
- Implementar la arquitectura IoT diseñada haciendo uso de robots móviles con ruedas en una aplicación colaborativa.

1.5. Estructura del Documento

El presente proyecto consta de 5 capítulos, el Capítulo 1 expone los estudios previos desarrollados tanto en el país como a nivel internacional acerca del tema planteado, así mismo presenta la identificación de la problemática dando paso a la justificación e importancia de la misma, a partir de ello se plantea el alcance del proyecto y los objetivos para lograr su desarrollo.

El Capítulo 2 menciona fundamentos teóricos acerca de los conceptos relacionados con el proyecto, entre los que se encuentran el origen del IoT, las plataformas disponibles para su desarrollo y las definiciones relacionados con la robótica móvil y colaborativa.

El Capítulo 3 expone el diseño del sistema Multi-Robot, la arquitectura IoT y su integración mediante un planificador automático. Así mismo, el capítulo 4 describe las pruebas realizadas y los resultados obtenidos producto de la experimentación, comparación y medición de variables.

En el Capítulo 5 se detallan las conclusiones obtenidas luego de la elaboración del proyecto y su implementación, las recomendaciones resultado del desarrollo y se proponen trabajos futuros basados en el proyecto.

CAPÍTULO 2

ESTADO DEL ARTE

El capítulo declara las bases científicas y tecnológicas con las cuales se desarrolló el proyecto. Se define a detalle las diferentes concepciones del Internet de las Cosas, su historia, aplicaciones, componentes y tipos de servicio. Además, se expone los conceptos referentes a robots móviles con ruedas, robots colaborativos y planificadores automáticos.

2.1. Internet de las Cosas

2.1.1. Definición

El término "Internet de las Cosas" no tiene una definición globalmente aceptada debido a la cantidad de conceptos que puede abarcar. A continuación, se expone los dos enfoques principales con los que se entiende y desarrolla el Internet de las Cosas, el primero por parte de Kevin Ashton (Ashton, 2009) y el segundo por parte de Cisco (Cisco, 2017).

Kevin Ashton es el cofundador de Auto-ID Center en el Instituto Tecnológico de Massachusetts, este centro se dedica a la investigación del IoT con especial enfoque en el uso de RFID. Según relata (Ashton, 2009) en su artículo, en 1999, después de ser contratado por la empresa P&G, se vio enfrascado en una investigación que buscaba dar una solución para la localización de los productos dentro de una tienda. En el mismo año realizó la presentación a la gerencia de P&G titulada: "Internet de las Cosas" acuñando el termino y consiguiendo la aprobación de los directivos para continuar con sus investigaciones. Kevin Ashton concibe el Internet de las cosas como una red en la que los dispositivos hacen uso del internet de forma independiente para informar su estado en todo momento y hacer uso de recursos en la nube. Ashton logró que más de 105 empresas patrocinaran las investigaciones del Auto - ID Center para desarrollar esta tecnología.

Cisco, una empresa internacional dedicada al campo computacional y de redes, a través de su división llamada: Cisco Internet Business Solutions Group (IBSG) y en conocimiento de las investigaciones desarrolladas en el Auto-ID Center, definió al IoT como un punto en la historia en el que más cosas que personas están conectadas a la red como se muestra en la Figura 1.

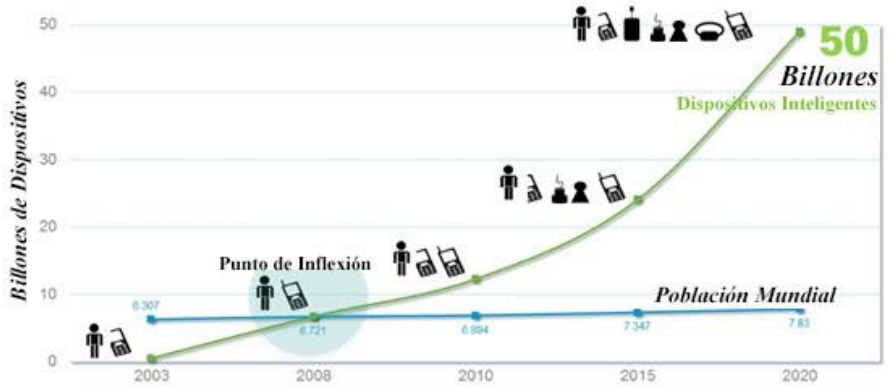


Figura 1 Definición de IoT según Cisco
Fuente: (Evans, 2011)

En base a lo expuesto por (Evans, 2011), en el año 2003 aún no existía el Internet de las Cosas debido a que la población mundial era aproximadamente 6.3 billones, mientras que los objetos conectados al internet eran alrededor de 500 millones, dando un estimado de 0.08 dispositivos por persona. En 2010 el número de artefactos conectados al internet creció drásticamente llegando a ser 12.5 billones frente a los 6.8 billones de personas en el mundo, dando un promedio de 1.84 dispositivos por persona. Gracias a estas cifras se pudo estimar el nacimiento del IoT entre el año 2008 y 2009. Se pronostica que para el año 2020 serán 50 billones los dispositivos conectados al internet mientras que la población mundial será de 7.8 billones.

2.1.2. Evolución

La definición del Internet de las Cosas que brinda Cisco define al IoT como el siguiente paso en el desarrollo del internet, a continuación, se describen las etapas principales que lo han llevado a su estado actual.

A. Una red de Computadoras

Desde las investigaciones iniciales para paquetería de datos en 1961 se buscó la forma de unir a computadoras en diferentes locaciones mediante una red que permita y garantice el intercambio de información. Como lo menciona en su artículo (Joskowicz, 2014), estas investigaciones se impulsaron con el apoyo militar y se creó ARPANET que en 1969 empezó a transportar sus primeros paquetes. Posteriormente se crearon nuevas redes, se diseñaron protocolos, infraestructuras y alianzas empresariales para crear el internet, el cual se concibe como la red de redes (Castells, 2000).

B. Computadoras y dispositivos móviles

Una vez que sistemas como *www* y protocolos como *http* se acogieron a nivel global junto al vertiginoso avance en microcontroladores y microprocesadores, el acceso a internet comenzó a ser posible para dispositivos más compactos. Como (Kantel, Tovar, & Serrano, 2010) lo mencionan en su artículo, la tercera generación del teléfono móvil permitió a sus usuarios tener acceso a internet, el cual evolucionó para poder unir a computadoras y dispositivos móviles en una sola red.

C. Computadoras, dispositivos móviles y sociedad

Los modelos que permitían a las personas conectarse entre sí evolucionaron. El clásico correo electrónico que permitió formar una red de contactos empezó a ser insuficiente. Según el artículo expuesto por (Qureshi, Mohammad AlManna, & Pasha Deshmukh, 2013) en 1995 se creó el sitio TheGlobe.com que permitía a sus usuarios compartir sus experiencias y entrar en contacto con otras personas con intereses similares. Lo cual abrió un nuevo horizonte, las relaciones sociales como parte del internet. En esta nueva etapa las interacciones entre personas, empresas y cualquier identidad social se vuelve parte intrínseca del internet.

D. Computadoras, dispositivos móviles, sociedad y cosas

Entre 2008 y 2009 se produce un punto de quiebre, los objetos conectados a internet superaron en número a la población mundial. La visión de Kevin Ashton y sus esfuerzos por crear un centro de investigación dedicado a la conexión de las cosas entre sí y con los seres humanos, el incremento exponencial de los dispositivos conectados y el creciente interés por parte de todos los sectores productivos y de desarrollo provocaron una nueva evolución: El internet de las Cosas.

E. Internet de Todo

El IoT supuso un enrome avance desde 1999. Sin embargo, empresas como Cisco no han frenado sus esfuerzos en la investigación y desarrollo de nuevas tecnologías. El siguiente paso consiste en conectar animales, plantas personas, procesos y datos. Lo cual permitiría que producciones de flores, crianza de ganado, mascotas y todo lo que rodea al ser humano se conecte al internet y se beneficie de sus características. Lo cual, según Pablo González el director de (Cisco, 2017), adquiere el nombre de Internet de Todo.

2.1.3. Impacto y avances en la robótica

La robótica es uno de los campos más afectados por el IoT. El artículo de (Guoqiang, Wee Peng, & Yonggang, 2012)expone los desafíos y aplicaciones de las arquitecturas para robótica en la nube proponiendo un modelo de computación elástica, donde los recursos se asignan dinámicamente para permitir el intercambio de información en aplicaciones robóticas. En el artículo propuesto por (Kehoe, Patil, Abbeel, & Goldberg, 2015) se enfoca los beneficios que representan para los robots y sistemas de automatización la infraestructura en la nube y sus de recursos. Entre los cuales se encuentran Big Data, Cloud Computing, Aprendizaje Colectivo y Computación Humana.

La aplicación de Cloud Robotics presentada por (Mohanarajah, Hunziker, D'Andrea, & Waibel, 2014) plantea el diseño y la implementación de Rapyuta, una plataforma de robótica en la nube de código abierto diseñada para aplicaciones robóticas multiproceso de gran ancho de banda, puede abarcar una gran variedad de escenarios robóticos y su protocolo de comunicación está basado en WebSockets. Los entornos informáticos de Rapyuta son privados, seguros y optimizados para el rendimiento de los datos que está determinado por la calidad de la conexión a la red.

2.1.4. Arquitectura IoT

Se conceptualizó a una arquitectura IoT como el conjunto de componentes físicos e intangibles que permiten a un grupo de dispositivos conectarse en una sola red sin necesitar a un ser humano para este propósito. A continuación, se describen los componentes que debe tener esta arquitectura para cumplir este objetivo y las tecnologías más comunes para su desarrollo.

A. Componentes

- ***Dispositivos***

Los objetos son el elemento principal de la arquitectura, el único requisito que deben cumplir es ser capaces de conectarse a la red sobre la que se esté desarrollando la arquitectura. Se presentan ejemplos de dispositivos que se han conectado a una arquitectura IoT con diferentes propósitos.

Uno de los principales lugares geográficos de desarrollo y producción influenciado por el IoT es la ciudad española de Santander. La ciudad fue seleccionada por la Comisión Europea para desarrollar el modelo de ciudad inteligente. Como (Espada Recarey, 2014) lo menciona en su artículo, se han instalado alrededor de 20.000 dispositivos en la ciudad como sensores, paneles informativos, repetidores móviles, etc., como se muestra en la Figura 2. Un sistema compuesto por más de 1600 sensores de los cuales 325 se encuentran bajo el asfalto. En el 2011 se realizó la

puesta en marcha del primer panel con información en tiempo real de los estacionamientos disponibles de la catedral, el control de estacionamientos de carga y descarga, paradas de autobús y vehículos para discapacitados.

Figura 2 Santander Smart City
Fuente: (CIC, 2013)

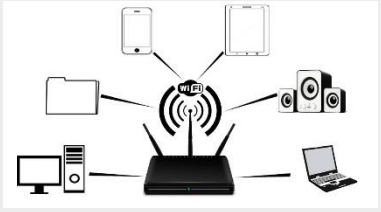
Por otra parte, Pachube es una plataforma virtual creada para gestionar datos que se generan alrededor del mundo en tiempo real, actualmente conocida como Xively, (Haque, 2012). Según lo relatan (Köhler, Wörner, & Wortmann, 2014), todo parte de un dispositivo denominado contador Geiger, el cual se encontraba localizado en una plataforma ubicada físicamente en Kyoto, el cual proporcionaba un *feed* de datos referentes a los niveles de radiactividad. Se volvió popular debido a que, días después del tsunami en Japón, la gente acudía a ver los datos que emitía el Geiger. El problema era que el contador se encontraba debajo de unos cuantos muros y quizá apuntando en una dirección errónea. La gente comenzó a conectar sus propios Geiger a la web mediante Pachube llegando a ser 72.000 *feeds* en pocos días, como se puede observar en la Figura 3. La información se mostraba en términos de consecuencias para la salud y radiación de fondo. Desde entonces Pachube ha liderado este movimiento de colaboración abierta que genera aplicaciones a la medida.

Figura 3 Pachube
Fuente: (Haque, 2012)

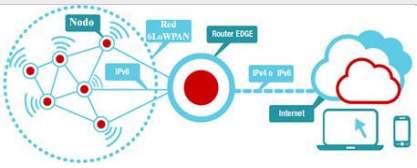
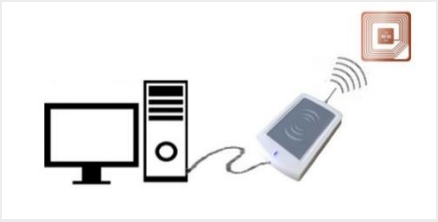
- ***Redes Inalámbricas***

Los dispositivos, en su mayoría móviles, necesitan un medio que permita su comunicación. Las redes inalámbricas permiten lograr este propósito. Las redes principales y sus características se muestran en la Tabla 1.

Tabla 1
Redes Inalámbricas

Red	Características	Ilustración
WiFi	Consiste en un adaptador Wireless ubicado en un dispositivo que traduce los datos a señales de radio y los transmite usando una antena, un router Wireless recibe esta señal y la decodifica para enviar dicha información al Internet usando una conexión física de cable. (Lozano Feliú, 2017)	
Bluetooth	Tecnología de corto alcance que se caracteriza por su reducido consumo de energía para transmisión de pequeñas cantidades de datos. Está compuesto por dos partes principales, un dispositivo de radio encargado de modular y transmitir la señal, y un controlador digital. (Lozano Feliú, 2017).	

CONTINÚA →

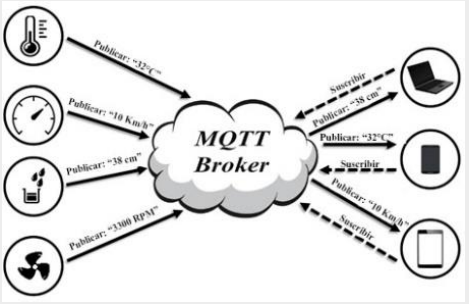
Zigbee	Tecnología que involucra un conjunto de protocolos de alto nivel de comunicación basada en el estándar IEEE 802.15.4 de redes de área personal para domótica. Diseñado para reemplazar la gran cantidad de sensores con la inclusión de una antena integrada, una batería y control de frecuencia. (Glen & Moreno, 2012).	
6LoWPAN	Tecnología de bajo consumo de red, diseñada en una topología de malla donde cada uno de sus nodos tiene su propia dirección IPv6. (Castillo Merchán, 2016).	
RFID	Tecnología que surgió con el propósito de identificación de objetos. Consiste en 3 elementos: Lectores, antenas y etiquetas. El lector envía una serie de ondas de radio a la etiqueta, la cual las capta a través de su microantena y activa un microchip, (Nava Díaz, Chavarría Juarez, Hervás Lucas, & Bravo Rodriguez, 2009).	

- ***Protocolos***

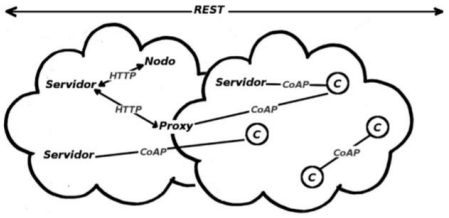
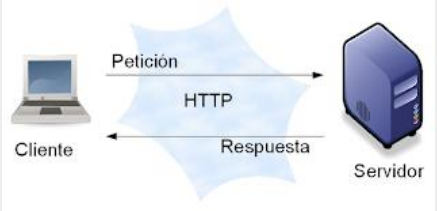
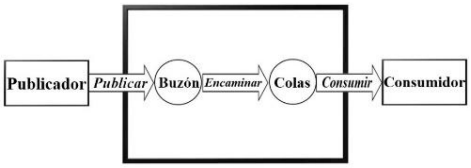
Los protocolos permiten que los dispositivos conectados a la red intercambien información, los cuales dependen de la red inalámbrica sobre la que se ha desarrollado la arquitectura IoT. En la Tabla 2 se mencionan los protocolos más comunes en los desarrollos de Internet de las Cosas.

Tabla 2

Protocolos de comunicación

Protocolo	Características	Ilustración
MQTT	Protocolo de código abierto creado por IBM destinado a la conectividad Machine-to-Machine (M2M). Se orienta a la comunicación de sensores debido a su bajo consumo energético y ancho de banda. Su arquitectura tiene topología de estrella con un nodo central llamado bróker y se basa en un modelo de publicación/suscripción. (Zambrano, Pérez, Plau, & Esteve, 2015).	

CONTINÚA →

CoAP	Protocolo a nivel de capa de aplicación usado en sensores de baja potencia. Especializado en el uso de nodos inalámbricos restringidos y limitados. Su modelo de interacción es cliente/servidor, (Castro Heredia, 2014).	
HTTP	Protocolo que sigue un modelo cliente-servidor. Una petición es iniciada por un cliente, y el servidor responde por medio de un mensaje que contiene el estado de la operación y su posible resultado, (Primo Guijarro, 2012).	
AMQP	Protocolo de estándar abierto para la comunicación mediante middleware orientado a mensajería. Consiste en la transferencia de mensajes de forma segura, asíncrona y confiable entre dos dispositivos (Rabelo, González, Pérez, & Delgado, 2015).	

B. Tipos de servicio

El internet de las cosas y su influencia en importantes áreas de desarrollo y producción ha provocado que diferentes empresas busquen formas de facilitar y monetizar los recursos necesarios para su aplicación. El conocer estos tipos de servicio y las empresas principales en cada modelo facilita la implementación de proyectos basados en IoT en función de su alcance y sus proyecciones a futuro. En la Figura 4 se muestra los tres tipos de servicio y ejemplos de empresas que lideran estos servicios.

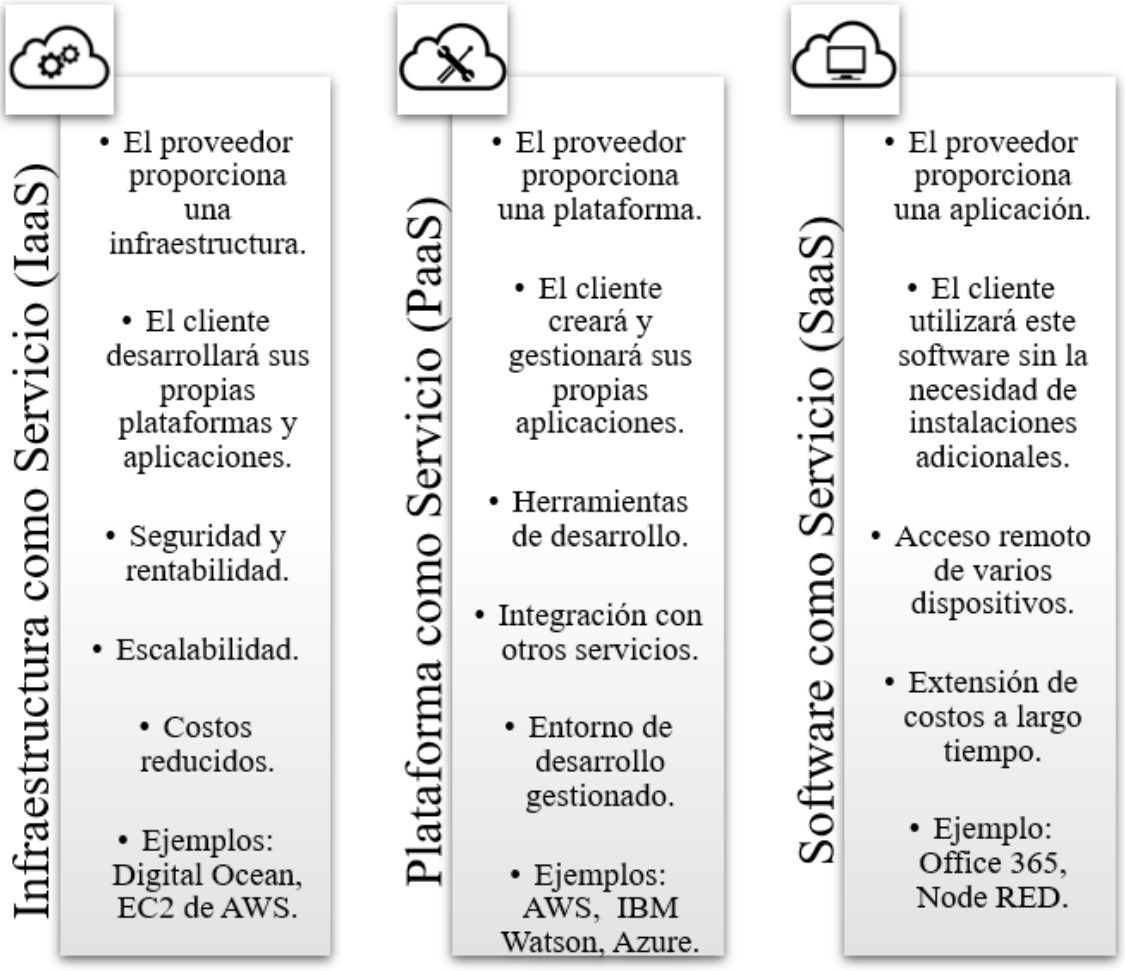


Figura 4 Comparación entre servicios de la Nube

2.2. Robots móviles con ruedas

2.2.1. Definición

Un robot móvil es una plataforma mecánica dotada de un sistema de locomoción, un sistema de sensores y un sistema de control. Dichos componentes se conjugan para permitir al robot desenvolverse dentro de un ambiente de trabajo con determinadas características. Cuando el sistema de locomoción hace uso de ruedas, los robots móviles con ruedas se abrevian como RMR.

Entre las aplicaciones basadas en RMR según (Cruz, 2008) se encuentra la aspiradora Scooba de iRobot que se muestra en la Figura 5.a) capaz de limpiar el suelo sin mezclar el líquido limpio

con el agua sucia. Por otra parte, Verro es un robot móvil diseñado con la finalidad de limpiar piscinas en un tiempo aproximado de 60 a 90 minutos, el cual se muestra en la Figura 5.b). En la Figura 5.c) se muestra a Hospi, un robot enfermero desarrollado por la empresa Matsishita equipado con sensores y programado con los datos del hospital para asistencia de pacientes.

Figura 5 Ejemplos RMR a) Scooba b) Verro c) Hospi

2.2.2. Componentes

Los sistemas de locomoción, sensores y control son los componentes principales del robot móvil, en esta se sección se expone sus tipos y funciones principales.

A. Sistemas de Locomoción

El sistema de locomoción del robot móvil es el encargado de dotarlo de movilidad, para este propósito se hace uso de varias configuraciones, a continuación, se hará una breve mención de los sistemas de locomoción en general.

- ***Tipos***

El tipo de sistema de locomoción del robot móvil está íntimamente ligado a su entorno de trabajo. Existen diferentes sistemas para los robots terrestres, acuáticos y aéreos. La Tabla 3 presenta los sistemas de locomoción típicos.

Tabla 3
Sistemas de locomoción típicos

	Sistema	Forma de movimiento	Ejemplo
Terrestres	Deslizantes	Simulando el deslizamiento de serpientes y gusanos.	Serpiente robótica diseñada y construida por (San Millán Rodríguez, 2012) mostrada en la Figura 6.a).
	Rodantes	Mediante ruedas.	Mini-robot móvil de tracción diferencial propuesto en el trabajo de (Ramírez, 2003) y mostrado en la Figura 6.b).
	Caminantes	Mediante patas que permite subclasificarlos en bípedos, hexápodos, cuadrúpedos, etc.	Nao, robot humanoide programable y autónomo desarrollado por Aldebaran Robotics y mostrado en la Figura 6.c).
	Rulantes	Mediante orugas o cadenas.	Robot explorador Arcangel diseñado por (Arévalo & Pino, 2005) y mostrado en la Figura 6.d).
Acuáticos	Flotante	Mediante turbinas o simulando el movimiento de serpientes	Prototipo recolector de material plástico flotante en el agua creado por (Montoya, Pérez, Garnica, Salamanca, & Simanca, 2012) y mostrado en la Figura 7.a)
	Submarino	Mediante propulsores o simulando las aletas de animales acuáticos.	Robot REMO I diseñado para la medición y observación oceanográfica por (Álvarez, Aracil, & García, 2009), mostrado en la Figura 7.b).
Aéreos	Ala rotativa	Mediante el impulso generado por la rotación de las alas	Dron Phantom 3 de la empresa expuesto en el trabajo de (Puertas, 2016) y mostrado en la Figura 8.a).
	Ala Fija	Mediante el impulso del aire debajo de las alas.	Prototipo de UAV multirotor de mini escala con estructura aerodinámica de ala fija diseñado por (Orbea & Moposita, 2017) y mostrado en la Figura 8.b).

Fuente: (Cruz, 2008)

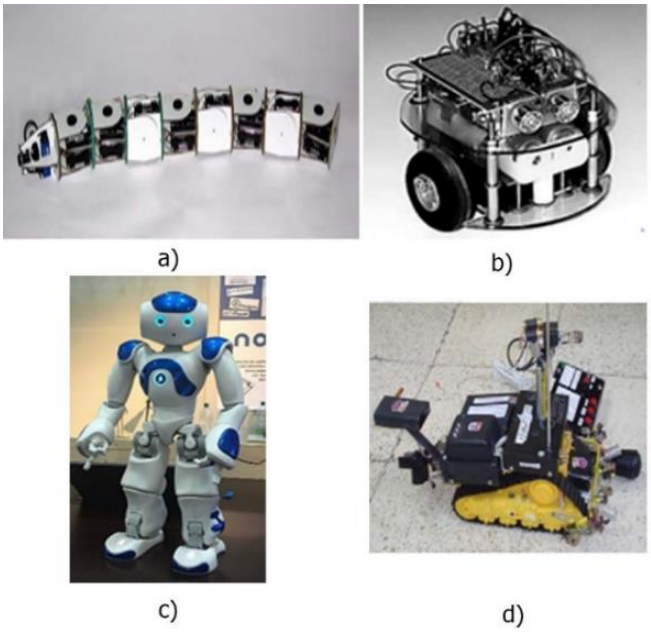


Figura 6 a) Serpiente robotica, b) Robot rodante, c) Nao, d) Robot explorador arcangel.

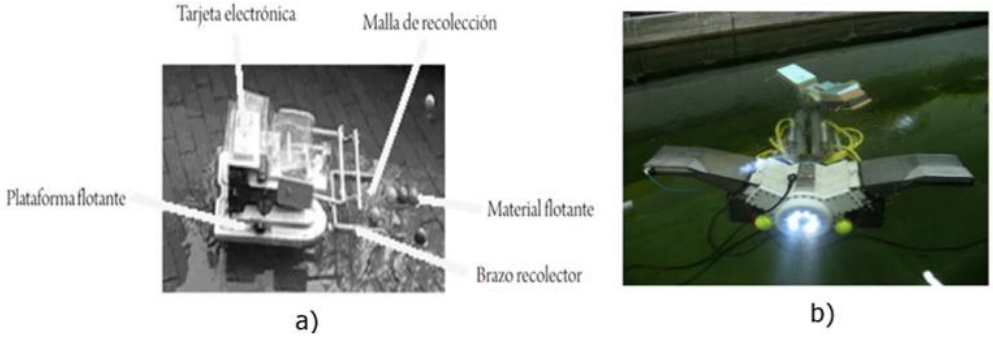


Figura 7 a) Flotante, b) Remo

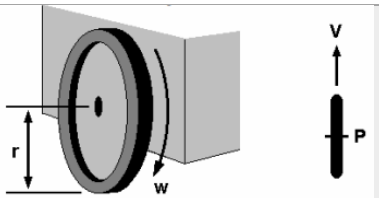
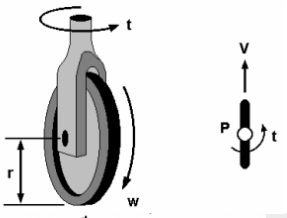
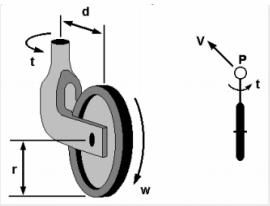
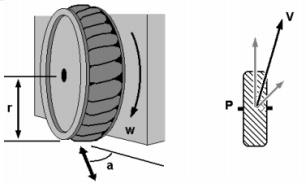
Figura 8 a) Dron Phantom, b) UAV

- ***Ruedas***

El tipo de ruedas y su configuración influyen en las prestaciones, el control y las capacidades de los robots móviles. La Tabla 4 expone los tipos principales de ruedas.

Tabla 4

Tipos de Ruedas

Nombre	Características	Ilustración
Rueda Fija	Recibe el nombre de rueda fija aquella cuyo eje está fijo a la estructura del robot. Esta configuración se usa generalmente para ayudar a la tracción del robot.	
Rueda orientable centrada	Su movimiento es la rotación alrededor del eje vertical que pasa a través del centro de la rueda. Es usada tanto como rueda de dirección como rueda de tracción.	
Rueda orientable no centrada	Conocida como rueda castor o rueda loca, su movimiento es la rotación alrededor de un eje vertical que no pasa a través del centro de la rueda. Se usa como rueda de estabilización o de dirección	
Rueda sueca	Conocida como rueda omnidireccional, gracias a los rodamientos montados en la superficie de la rueda puede desplazarse en dirección perpendicular al plano de la rueda.	

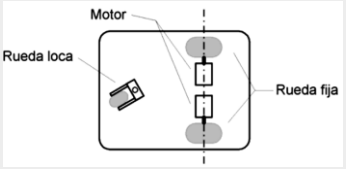
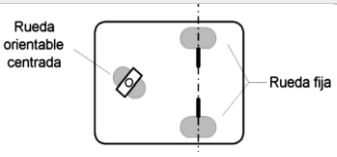
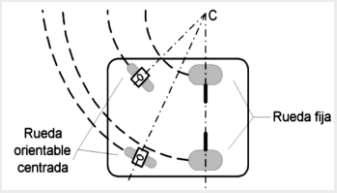
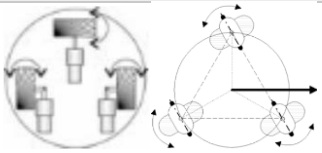
Fuente: (Cruz, 2008)

- ***Configuración Cinemática***

La disposición de las ruedas en la estructura del robot junto con el tipo de rueda a usarse, dan lugar a diferentes configuraciones cinemáticas de RMR. La Tabla 5 detalla las configuraciones cinemáticas principales en estos robots.

Tabla 5

Configuraciones Cinemática

Configuración	Características	Ilustración
Uniciclo	En esta configuración cinemática el robot tiene dos ruedas fijas con control independiente para tracción y dirección, y una rueda loca sin control para brindar estabilidad al robot. Esto permite independizar la velocidad angular y lineal.	
Triciclo	En esta configuración cinemática el robot tiene dos ruedas fijas sobre un mismo eje sin control y una rueda centrada orientable encargada de la tracción y dirección.	
Cuatriciclo	En esta configuración cinemática el robot posee dos ruedas centradas orientables y dos ruedas fijas, de este modo se puede hacer uso de la configuración Ackerman para el direccionamiento. Esta estructura brinda mayor estabilidad y evita el deslizamiento en las ruedas.	
Omnidireccional	En esta configuración cinemática el robot es capaz de moverse en cualquier dirección sin necesidad de reorientarse perpendicular. Esto se logra ya sea con una configuración de cuatro ruedas omnidireccionales o ruedas orientables centradas.	

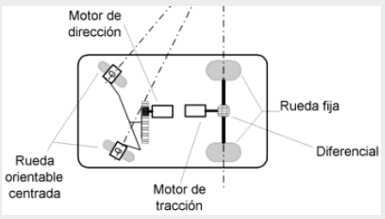
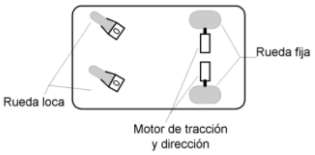
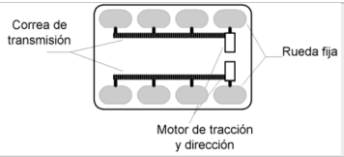
Fuente: (Cruz, 2008)

- ***Tracción y Dirección***

En función del tipo de tracción y dirección se puede dotar de maniobrabilidad, tracción, capacidad de carga, etc., al robot. En la Tabla 6 se resumen los tipos de tracción y dirección principales en los RMR.

Tabla 6

Tipos de Tracción y Dirección

Tipo	Características	Ilustración
Tracción y dirección en ejes independientes.	En este sistema se efectúa la tracción en las ruedas traseras y la dirección en las ruedas delanteras o viceversa. Esto facilita el cambio de dirección, pero su precisión depende de la fricción entre el piso y las ruedas, además el radio de giro es mayor que en otros sistemas.	
Tracción y dirección en un mismo eje.	También llamada tracción diferencial, esta tracción se consigue con un control independiente en las ruedas del mismo eje Y ruedas locas en los demás ejes.	
Tracción y dirección sobre todos los ejes.	Su aplicación está destinada a terrenos hostiles. En este sistema se tiene control de tracción y dirección sobre todos los ejes.	

Fuente: (Cruz, 2008)

B. Sistema de Sensores

El sistema de sensores permite al robot móvil recibir datos de su entorno y reaccionar según su algoritmo de control, en la mayoría de las aplicaciones de RMR estos sensores se complementan entre sí para mejorar sus estimaciones.

- ***Tipos***

Según (Cruz, 2008) los sensores en robótica se pueden clasificar en función de los criterios mostrados en la Tabla 7:

Tabla 7

Sensores en robótica

Sensores en robótica			
Según el medio relativo al robot	***Según el tipo de interacción robot-objeto***	***Según el tipo de información***	***Según el tipo de funcionamiento***
Propioceptivos	Contacto	Elementales	Carga eléctrica
Exteroceptivos	No contacto	Complejos	Radiación luminosa
			Resistencia
			Otros

En la Tabla 7 se definieron cuatro criterios para la clasificación de los sensores en robótica. Según el medio relativo pueden ser propioceptivos y exteroceptivos. Los primeros están destinados para medir variables internas del robot, como corriente, velocidad, aceleración, etc. Los segundos están ubicados de forma que miden una variable del entorno del robot como temperatura ambiente, distancia, color, etc.

El segundo criterio de clasificación se basa en la interacción que tiene el robot con el objeto a medir. Si el sensor necesita estar en contacto físico con el objeto, por ejemplo, un caudalímetro de turbina, un LM35, un PT100, etc., el sensor es de contacto. De lo contrario, el sensor es de no contacto, por ejemplo, termómetro infrarrojo y caudalímetro ultrasónico.

El siguiente criterio se basa en el tipo de información que pueden proporcionar los sensores. Si el sensor provee una única señal de una magnitud específica, entonces el sensor es de tipo elemental, ejemplos de estos sensores son un termopar y un sensor infrarrojo de distancia. Si el sensor es capaz de transmitir su información mediante vectores o arreglos matriciales estos son de tipo complejo, por ejemplo, cámaras de video. El último criterio de clasificación de los sensores se basa en su principio de funcionamiento. Las variaciones en la resistencia del material, carga eléctrica, capacitancia, inductancia, etc.

- ***Estimadores***

La navegación de un robot móvil, según (Leonard & Durrant-Whyte, 1991) debe resolver tres interrogantes principales: “¿Dónde estoy?”, “¿A dónde voy?” y “¿Cómo debo llegar ahí?”. Los datos provenientes de los sensores deben permitirle al robot estimar su posición y orientación, mantener el mapa del entorno y detectar los posibles obstáculos.

Para que el RMR sea capaz de ejecutar las tareas primordiales, es necesario poder determinar su localización y orientación con respecto a un sistema de referencia absoluto. Lo que significa

hallar las coordenadas *x*, *y*, *z* y los ángulos de giro *φx*, *φy*, *φz* del sistema de coordenadas móvil solidario al robot con respecto al sistema fijo.

El estudio se enfocará en un plano lo que reduce el problema a encontrar las coordenadas *x*, *y* y el ángulo de rotación *φ* alrededor del eje *z*. En la Figura 9 se muestran las variables cinemáticas mencionadas.

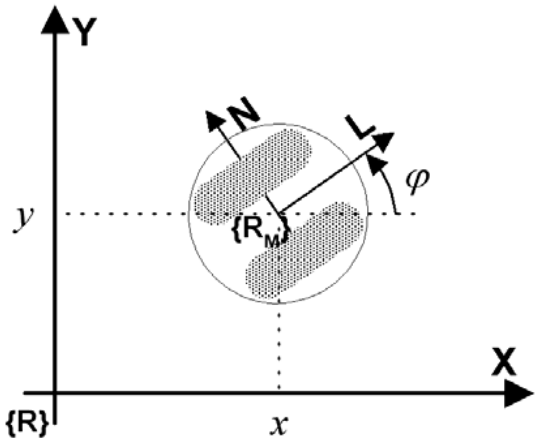


Figura 9 Variables cinemáticas RMR
Fuente: (Cruz, 2008)

Generalmente los RMR están provistos de codificadores en los ejes de movimiento que permiten estimar su localización en el tiempo. Sin embargo, la medición de la posición con este método está sujeta a pequeños errores sistemáticos que se van acumulando conforme el robot se mueve. Los estimadores de posición y orientación más comunes fueron clasificados de forma sencilla por parte de (González Jiménez & Ollero Baturone, 1996), en su clasificación se menciona a los estimadores basados en medidas internas que se muestran en la Tabla 8.

Tabla 8
Estimadores basados en medias internas

Odometría	• Sensores Doppler • Codificadores
Navegación Inercial	• Giroscopios • Acelerómetros

Fuente: (Cruz, 2008)

Este tipo de estimadores buscan obtener una aproximación de la posición y orientación del robot haciendo uso de medidas de variables internas, cómo número de vueltas y la velocidad angular de las ruedas, cambios en la dirección y sentido del movimiento, aceleraciones, velocidades, etc. En función al tipo de datos usados para estas estimaciones, se puede clasificar a estos sistemas en odométricos y mediante navegación inercial.

- **Sistemas Odométricos**

La odometría es un sistema simple que se basa en el conteo del número de vueltas dadas por las ruedas del robot para estimar su posición y orientación. La medición se ve afectada por la resolución del sensor, el cambio en el radio efectivo de las ruedas debido al desgaste, el deslizamiento por la fricción entre las ruedas y el piso, las irregularidades del suelo y las variaciones en el peso del robot cuando este transporta carga, sin embargo, estas dificultades no evitan que sea un sistema usado con frecuencia, debido a su bajo costo, simple implementación y altas tasas de muestreo. En la Figura 10 se muestra el movimiento de un robot móvil con ruedas y las variables cinemáticas asociadas al desplazamiento.

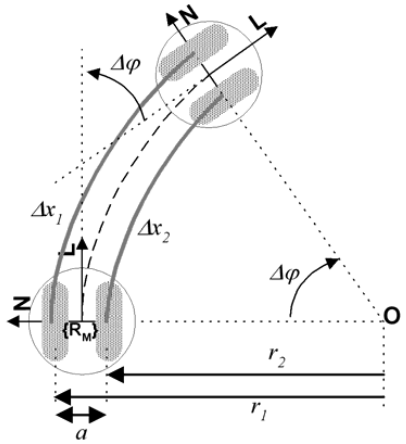


Figura 10 Variables cinemáticas del sistema odométrico
Fuente: (Cruz, 2008)

El movimiento de la rueda izquierda y derecha se obtienen con la ecuación (2-1) y (2-2), respectivamente.

$$\Delta x_1 = r_1 \cdot \Delta\varphi \qquad (2\text{-}1)$$

$$\Delta x_2 = r_2 \cdot \Delta\varphi \qquad (2\text{-}2)$$

El punto medio del robot, en donde se ubica el sistema de referencia móvil R_M, se define por la ecuación (2-3).

$$\Delta x = \frac{\Delta x_1 - \Delta x_2}{2} \qquad (2\text{-}3)$$

Para el cambio de trayectoria se estima el cambio de la pendiente como se muestra en las ecuaciones (2-4) y (2-5).

$$\Delta\varphi \cdot (r_1 - r_2) = (\Delta x_1 - \Delta x_2) \qquad (2\text{-}4)$$

$$\Delta\varphi = \frac{(\Delta x_1 - \Delta x_2)}{(r_1 - r_2)} = \frac{(\Delta x_1 - \Delta x_2)}{a} \qquad (2\text{-}5)$$

Se utilizan encoders o codificadores ópticos incrementales acoplados al menos a dos ruedas del robot. El principio de funcionamiento de los encoders ópticos consiste en el conteo de ranuras en un disco que gira solidario al eje de la rueda, de este modo, cierto número de pulsos representa una cantidad determinada de desplazamiento angular. En la Figura 11 se muestra un codificador óptico.

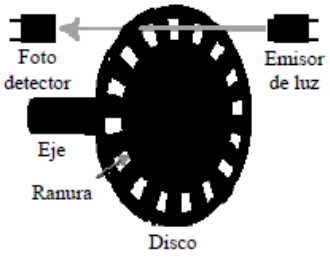


Figura 11 Codificador Óptico
Fuente: (Cruz, 2008)

- **Navegación Inercial**

Los métodos para realizar la navegación inercial se basan en el uso de sensores inerciales: Acelerómetros, giroscopios y magnetómetros. Estos permiten determinar aceleración lineal, velocidad angular y orientación con respecto al norte geográfico, respectivamente. Actualmente estos sensores se han integrado en unidades de medición inercial de bajo costo y simples de instalar.

La dificultad en la navegación inercial reside en el tratamiento de los datos y la calibración de los sensores. La relación señal/ruido debe ser adecuada para medir magnitudes pequeñas, la ubicación del sensor debe coincidir con los ejes de medición del robot, pequeños errores en sus mediciones se incrementan en los procesos de integración numérica. La navegación inercial se ve menos afectada por el entorno en comparación a la odometría. Los errores producidos por las mediciones son acumulativos. En la siguiente ilustración se muestra un giroscopio de 3 ejes que permite la medición de la velocidad angular y la temperatura ambiente.

Figura 12 Giroscopio de 3 ejes
Fuente: (Digillent, 2016)

C. Sistema de Control

El sistema de control del RMR es el encargado de interpretar los datos provenientes del sistema de sensores y convertirlos en acciones para el sistema de locomoción y actuadores. El nivel de independencia de un operador humano en este sistema permite clasificar a los RMR en dos tipos teleoperado y autónomo. Un RMR teoleoperado es aquel que necesita de un operador humano que controle sus acciones mediante un mando a distancia. Un RMR autónomo es aquel capaz de

ejecutar sus acciones con cierto grado de independencia. Las tareas del sistema de control pueden extenderse en función de las capacidades del robot como la comunicación, gestión de alarmas, ejecución de secuencias, verificación de tareas, etc.

2.3. Robots Colaborativos

Los robots que trabajan en busca de lograr un objetivo en común reciben el nombre de robots colaborativos. El conjunto de robots es un sistema Multi-Robot y según (Oussama Khatib, 2008) las principales razones para su estudio son:

- La tarea por realizar tiene una complejidad muy elevada para un solo robot.
- La construcción de varios robots de recursos limitados es más sencilla que la construcción de un solo robot complejo.
- La naturaleza de la tarea necesita de un trabajo colaborativo.
- Se puede usar el paralelismo de tareas para resolver el problema más rápidamente.
- Se puede incrementar la robustez del sistema mediante la redundancia.

2.3.1. Tipos

Según (Parker, 2012) se puede clasificar a los sistemas Multi-Robot según el mecanismo de interacción entre los robots cuyas características se resumen en la Tabla 9.

Tabla 9
Clasificación de los sistemas Multi-Robot según su mecanismo de interacción

Tipo	Características
Colectivo	Los robots son simples y no están al tanto de los demás miembros del sistema, sin embargo, todos cumplen metas comunes.
Cooperativo	Los robots tienen un grado de conciencia sobre el estado del resto del equipo. Sus resultados contribuyen al objetivo global.
Colaborativo	Los robots están conscientes de los demás miembros del equipo y a pesar de no tener los mismos objetivos, están dispuestos a ayudar a los demás miembros en sus objetivos individuales.
Coordinación	El objetivo principal de cada robot es minimizar la interferencia con el resto de los miembros de equipo.

2.3.2. Arquitectura

Una arquitectura para un sistema Multi-Robot hace referencia a la forma en que interactúan, funcionan y se organizan entre si los distintos miembros del sistema (Ruiz Libreros, 2013). Existen dos arquitecturas principales para sistemas multi-robot centralizada y descentralizada.

A. Centralizada

En esta arquitectura existe un agente que tiene el control de todos los robots del equipo. Se requiere que dicho agente tenga conocimiento global del entorno y los requerimientos a cumplir. El agente central gestiona las tareas y las distribuye entre los robots como los mostrados en la Figura 13.

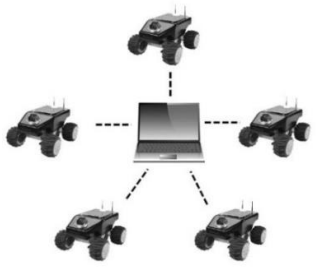


Figura 13 Arquitectura Centralizado

B. Descentralizada

En esta arquitectura los robots perciben el entorno y toman sus propias decisiones sin necesidad de un agente central que determine su funcionamiento. Dentro de esta arquitectura se pueden distinguir dos tipos jerárquicas y distribuida como se muestra en la Figura 14.

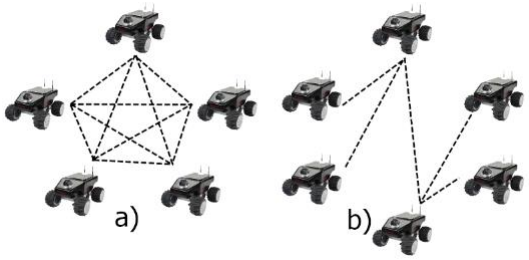


Figura 14 Arquitectura Descentralizada: a) jerárquica, b) distribuida.

En la arquitectura jerárquica los robots trabajan en distintas etapas del problema en función de sus capacidades. En la arquitectura distribuida los robots subdividen el problema en tareas más sencillas, una vez finalizado cierto nivel de complejidad se procede a realizar el siguiente nivel.

2.3.3. Planificación Automática

Según (Vera Cañal, 2015), la planificación es parte de la Inteligencia Artificial cuyo objetivo principal es producir planes, generalmente diseñados para ejecutarse sobre un robot u otro agente. Según (Ghallab & Nau, 2004) la planificación es el proceso por el cual se escogen y organizan tareas anticipando sus resultados esperados, de forma abstracta y explícita, para lograr determinados objetivos previamente definidos. La definición de estos componentes de la planificación determina su nivel de dificultad. Los problemas que se pretenden resolver mediante un planificador automático permite clasificarlos de acuerdo a (Vera Cañal, 2015) como se muestra en la Tabla 10.

Tabla 10

Clasificación de los planificadores automáticos según el problema a resolver

Tipo	**Características**
Clásico	Es el planificador más sencillo, se basa en un estado inicial con un único agente que realiza acciones determinísticas con independencia del tiempo y que se deben ejecutar en secuencia lineal. Lo cual permite que se pueda determinar el estado preciso del proceso después de cada acción.
Proceso de decisión de Markov	Se abrevia como MDP por sus siglas en inglés, este planificador es destinado para tareas más complejas en las que los resultados de cada acción son parcialmente aleatorios debido a que los agentes realizan acciones no deterministas. Sin embargo, la observabilidad del sistema debe ser completa.
MDP Parcialmente observable	En este planificador se hace uso de un MDP para la dinámica del sistema, sin embargo, no se puede observar directamente el estado de los agentes, por lo que estos se deben determinar mediante una distribución de probabilidad sobre el conjunto de estados posibles.
Planificación Multi-agente	En este planificador se busca la coordinación de recursos y acciones de varios agentes, este tipo de planificación depende de la comunicación entre ellos.

CAPÍTULO 3

DISEÑO E IMPLEMENTACIÓN

El capítulo presenta el diseño e implementación de la arquitectura IoT para robótica colaborativa. La metodología de diseño se basó en la norma (VDI-2206, 2014) para sistemas Mecatrónicos. Por lo cual, se procedió a determinar los requerimientos del proyecto por parte del Laboratorio de Mecatrónica y Sistemas Dinámicos de la universidad, posteriormente se diseñó los subsistemas Multi – Robot y Arquitectura IoT. Finalmente, se integró los subsistemas mediante un planificador automático que se encarga del control colaborativo de los robots.

3.1. Metodología

La norma (VDI-2206, 2014) conceptualiza el diseño para sistemas mecatrónicos basándose en los requerimientos que debe cumplir el producto, se procede a una primera etapa conceptual en donde se definen subsistemas con determinadas funciones a cumplir, los cuales se desarrollan de forma específica en la siguiente fase de diseño.

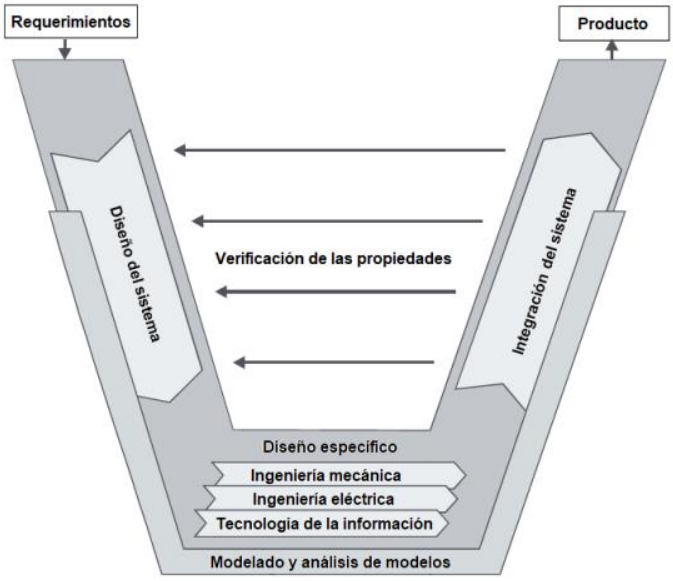


Figura 15 Modelo en V de Diseño Mecatrónico
Fuente: (VDI-2206, 2014)

Posteriormente se integra el sistema y se realizan pruebas para verificar sus propiedades. En la Figura 15 se muestra el proceso de diseño definido por la norma VDI 2206 y a continuación se da una breve descripción de su aplicación en el proyecto.

3.1.1. Requerimientos

Los requerimientos fueron definidos por el Laboratorio de Mecatrónica y Sistemas Dinámicos de la universidad. Los cuales se convirtieron en especificaciones objetivo y se dividieron en funciones que deben cumplir cada subsistema.

3.1.2. Diseño del Sistema

En esta etapa se conceptualizó las diferentes opciones para cada subsistema. Se eligió las alternativas que cumplieron con las características deseadas. La evaluación se realizó mediante matrices de selección siguiendo el procedimiento detallado por (Ulrich & Eppinger, 2013).

3.1.3. Diseño Específico

Los conceptos seleccionados para los subsistemas se diseñaron haciendo uso de herramientas matemáticas que permitieron caracterizar y controlar su comportamiento.

3.1.4. Integración del Sistema

La integración del sistema se realizó mediante un planificador automático que permitió la interacción de los subsistemas. El planificador automático también cumple la función de un control colaborativo centralizado para los robots.

3.1.5. Verificación de las propiedades

La verificación de las propiedades del sistema se detalla en el capítulo 4, la metodología se basa en evaluar criterios IoT, velocidad de respuesta, funcionalidad, costo computacional y comportamiento colaborativo.

3.1.6. Modelado y análisis de modelos

El modelado y análisis de modelos se desarrolló en el diseño específico y la integración del sistema, en donde se usó modelos que caracterizaron el comportamiento de los subsistemas y permitieron controlarlos.

3.1.7. Producto

El producto que se obtuvo es una arquitectura IoT para robótica colaborativa capaz de intercomunicar a los miembros del sistema Multi-Robot con la inclusión de un planificador automático que se encarga del control colaborativo en una aplicación demostrativa del sistema.

3.2. Subsistema Multi – Robot

3.2.1. Requerimientos

Los requerimientos para el subsistema Multi-Robot se muestran en la Tabla 11. La escala de 1 al 5 evalúa la prioridad de menor a mayor, respectivamente.

Tabla 11
Evaluación de requerimientos para el sistema multi-robot

Requerimiento	**Prioridad**
La navegación de los robots debe permitirlos ubicarse dentro de un entorno de trabajo de 3 metros de largo por 2 metros de ancho.	3
Los robots deben ser capaces de manipular un objeto con propósitos demostrativos, por lo cual no existe restricciones de peso ni forma.	4
Evitar choques frontales.	3
Hacer uso de los robots disponibles por laboratorio. Las características de su sistema de locomoción se presentan en la Tabla 12. Las características del sistema de sensores y actuador se presentan en la Tabla 13. En la Figura 16 se muestra los elementos de los robots ubicados en su estructura.	5

Tabla 12

Características del sistema de locomoción de los robots móviles

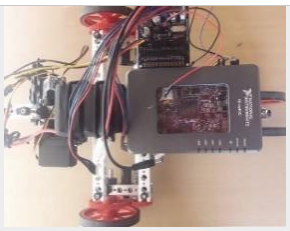
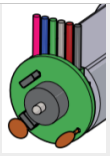
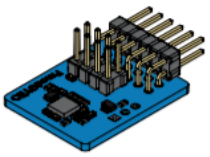
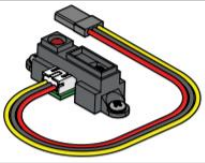
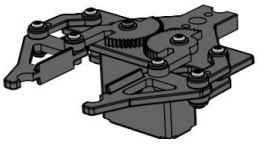
Característica	Detalle	Ilustración
Configuración cinemática	Uniciclo con tracción y dirección sobre el eje delantero y una rueda no orientable descentralizada para estabilización.	
Motores	Motores DC de 6 [V], y potencia de 2.88 [W]. (Sha Yang Ye, 2014)	
Caja de reducción	De tipo paralela con reducción de 1/52.734.	
Ruedas	Ruedas de plástico recubiertas de caucho con diámetro efectivo de 90 [mm]	
Engranaje final	Piñón de 40 dientes y corona de 80 dientes.	

Tabla 13

Características del sistema de sensores y actuador de los robots móviles.

Elemento	Características	Ilustración
Encoder	Encoder de 6 pulso por revolución mediante sensores de efecto hall.	

CONTINÚA →

Giroscopio	Giroscopio de 16 bits de resolución con entrada de -125 a 125 [dps].	
Sharp IR	Sensor infrarrojo para distancia basado en reflexión de luz.	
Gripper	Apertura de seis centímetros. Inclinación de 90º en Pitch.	
Controlador	Tarjeta de desarrollo NI MyRIO.	

Fuente: (National Instruments)

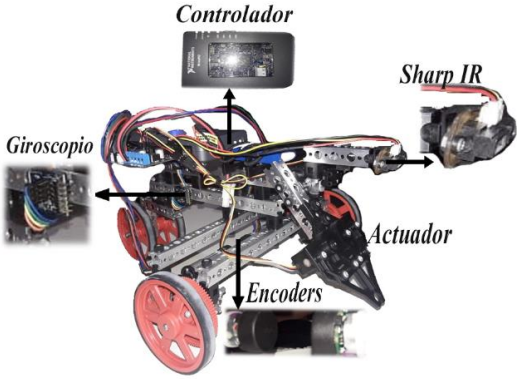


Figura 16 Elementos del Robot

3.2.2. Diseño

En función a los requerimientos definidos por el laboratorio se determinaron las características que debe cumplir el sistema de navegación, las cuales se muestran en la Tabla 14. El nivel de importancia de estas características se determinó mediante una matriz de selección dando los resultados presentados en la Tabla 15.

Tabla 14

Características del Sistema de Navegación

Característica	Observación
Simplicidad	Minimizar el procesamiento por parte del controlador del robot.
Precisión	Se debe garantizar una precisión del 80%
Exactitud	La exactitud en la posición debe ser de 80%
Costo energético	Minimizar el consumo energético debido al uso de baterías.

Tabla 15

Importancia de las Características del Sistema de Navegación

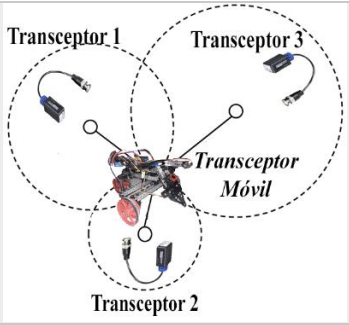
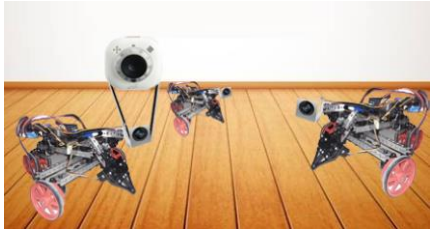
Característica	Importancia [%]
Simplicidad	30
Precisión	30
Exactitud	30
Costo energético	10

A. Elección del método de navegación

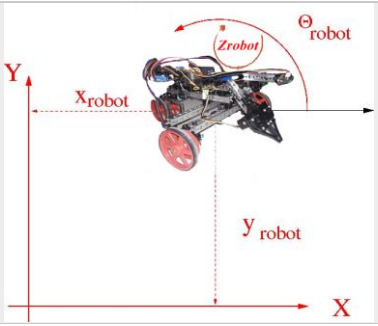
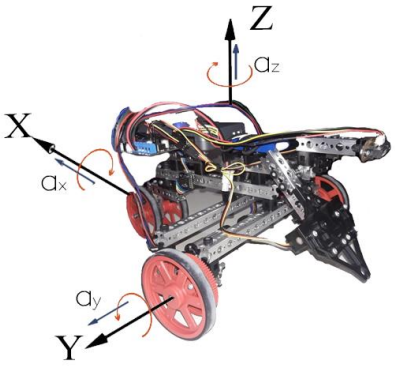
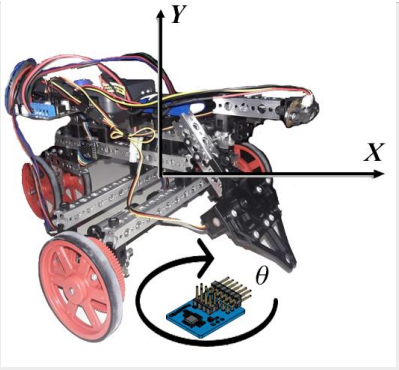
Para la selección del método de navegación se desarrollaron 3 posibles conceptos de solución que se muestran en la Tabla 16.

Tabla 16

Conceptos para la elección del sistema de navegación

Concepto	Ilustración
1) Torres de transmisión de ultrasonido para trilateración.	
2) Visión Artificial mediante cámaras de video montadas en los robots.	

CONTINÚA →

3) Odometría haciendo uso de los encoders disponibles en los motores.	
4) Sistema inercial mediante una IMU de seis grados de libertad.	
5) Sistema hibrido que haga uso de un giroscópico para el movimiento angular y odometría para el movimiento lineal.	

Se evaluaron los conceptos en función de las características presentadas en la Tabla 16. Los resultados se muestran en la Tabla 17.

Tabla 17

Evaluación de los conceptos para la elección del sistema de navegación

Concepto	Valoración
1	17.33%
2	21.67%
3	18.33%
4	14.33%
5	28.33%

El método de navegación que se acopla mejor a los requerimientos es el número cinco. El cual consiste en un sistema hibrido con navegación inercial para la orientación y odometría para la posición lineal.

B. Estimación de la posición

Una vez seleccionado el método de navegación, se procedió al desarrollo cinemático del robot como se muestra en la Figura 17. El punto de origen se fija en el encendido del robot.

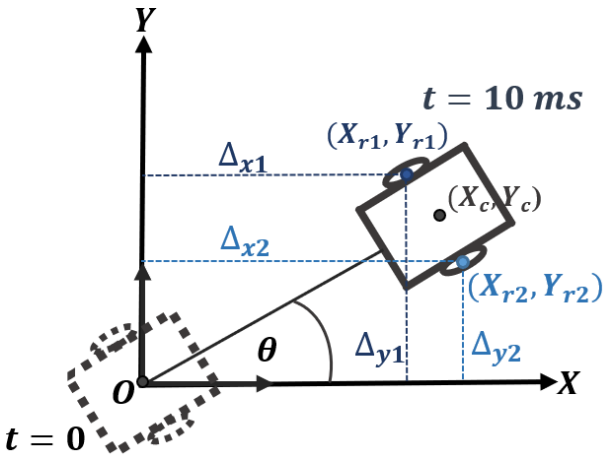


Figura 17 Estimación de la posición

En el tiempo *t = 0 [ms]* el robot se encuentra en el origen y ha girado un ángulo θ para el tiempo *t= 10 [ms]* el robot se ha movido del punto inicial *O* hasta el punto X_c, Y_c. La rueda izquierda se ha movido una distancia *Δx1* y la rueda derecha una distancia *Δx2*. Estos desplazamientos se obtienen mediante las ecuaciones (3-1) y (3-2).

$$\Delta x1 = k \cdot \Delta p_1 \cdot \sin(\theta) \qquad (3\text{-}1)$$

$$\Delta x2 = k \cdot \Delta p_2 \cdot \sin(\theta) \qquad (3\text{-}2)$$

En donde *Δp1* y *Δp2* corresponde al número de pulsos en 10 ms de los encoder de las ruedas izquierda y derecha respectivamente. *k* se calcula mediante la ecuación (3-3).

$$k = sign(g) \cdot \frac{\pi \cdot d \cdot N_p}{2 \cdot N \cdot N_c \cdot {r_c}^{-1}} \qquad (3\text{-}3)$$

En donde:

g: Sentido de giro de las ruedas para avanzar. Es negativo si el giro es horario y positivo si el giro es antihorario.

d: Diámetro de las ruedas en centímetros

N: Número de pulsos por revolución del encoder

Np: Número de dientes del piñón

Nc: Número de dientes de la corona

rc: Relación de transmisión de la caja reductora.

Reemplazando se obtiene:

$$k = -1 \cdot \frac{\pi \cdot 9 \cdot 40}{2 \cdot 6 \cdot 80 \cdot 52.734}$$

$$k = -0.0223$$

Para hallar la coordenada de las ruedas X_{r1} y X_{r2} se hace uso de las ecuaciones (3-4) y (3-5). En donde el subíndice *k* corresponde al instante de tiempo.

$$X_{r1_k} = X_{r1_{k-1}} + \Delta x1 \qquad (3\text{-}4)$$

$$X_{r2_k} = X_{r2_{k-1}} + \Delta x2 \qquad (3\text{-}5)$$

El punto del centro X_c se halla con la ecuación (3-6):

$$X_c = (X_{r1} + X_{r2})/2 \qquad (3\text{-}6)$$

La coordenada Y_c se halla mediante el mismo procedimiento al reemplazar $\sin(\theta)$ por $\cos(\theta)$ en las ecuaciones (3-4) y (3-5).

- **Desbordamiento del número de pulsos**

La variable en la que se almacenaron el número de pulsos del encoder tiene una longitud de 16 bits. Sus valores se encuentran entre los límites *–32 768* y *+ 32 767*. Sin embargo, el desbordamiento podría causar un cambio brusco de valores con lo cual Δp tendría una magnitud muy alta. Por lo cual, se condicionó la medición de forma que valores mayores a 100 en Δp son ignorados. En el proceso práctico los valores de Δp están en el orden de las unidades y un desbordamiento produciría valores en el orden de los miles. Mediante este condicionamiento, en caso de desbordamiento, se agrega un pequeño error por omisión de datos para evitar uno mayor por medición falsa.

C. Estimación de la orientación

La estimación del ángulo que ha girado el robot se realizó mediante el tratamiento de la señal del eje X del giroscopio. El proceso consistió en determinar valores iniciales, desplazar la señal al origen y eliminar valores dentro de una ventana de medición. Posteriormente se filtró la señal y se optimizó la convergencia a cero del filtro. En la siguiente etapa se integró numéricamente la señal y se solucionó el desbordamiento del ángulo. Finalmente se transformó a grados y se condicionó la activación del proceso de estimación.

- **Cálculo de valores iniciales**

Los datos iniciales del giroscopio presentan variaciones como las mostradas en la Figura 18. Las variaciones se deben a la temperatura ambiente, la posición inicial del giroscopio y otros factores ambientales que exigen un proceso dinámico para determinar el desplazamiento en magnitud O_f y la amplitud de las oscilaciones A_m.

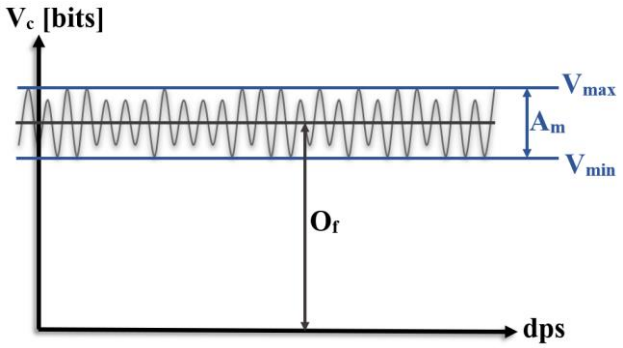


Figura 18 Representación de las oscilaciones del giroscopio

Para lograr este propósito se realizaron 150 mediciones iniciales m_i, de las cuales se determinó el valor máximo V_{max} y mínimo V_{min}, se calcula el offset O_f y la amplitud A_m con las ecuaciones (3-7) y (3-8).

$$O_f = \frac{\sum_{i=1}^{150} m_i}{150} \qquad (3\text{-}7)$$

$$A_m = V_{max} - V_{min} \qquad (3\text{-}8)$$

- **Desplazamiento al origen y ventana de medición**

Una vez que se determinaron los valore iniciales O_f y A_m, los valores crudos del giroscopio V_c se modificaron con O_f para obtener una señal desplazada al origen V_o mediante la ecuación (3-9) y como se muestra en la Figura 19.

$$V_o = V_c - O_f \qquad (3\text{-}9)$$

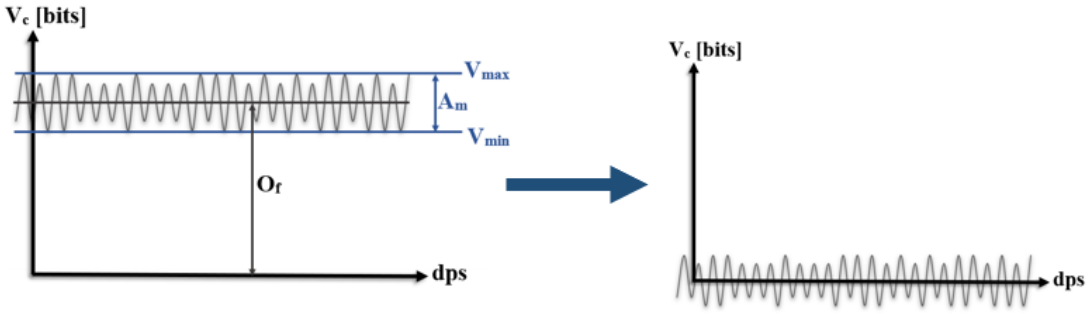


Figura 19 Desplazamiento de la señal del giroscopio al origen

Las pequeñas oscilaciones cuando el robot está inmóvil se eliminaron mediante una ventana de medición. De este modo se modificó V_o mediante A_m para obtener una señal inicial V_i que anula valores dentro de la ventana de medición como lo determina la ecuación (3-10) y se muestra en la Figura 20.

$$V_i = \begin{cases} V_o, & si\ |V_o| > 2 \cdot A_m \\ 0, & si\ |V_o| \leq 2 \cdot A_m \end{cases} \qquad (3\text{-}10)$$

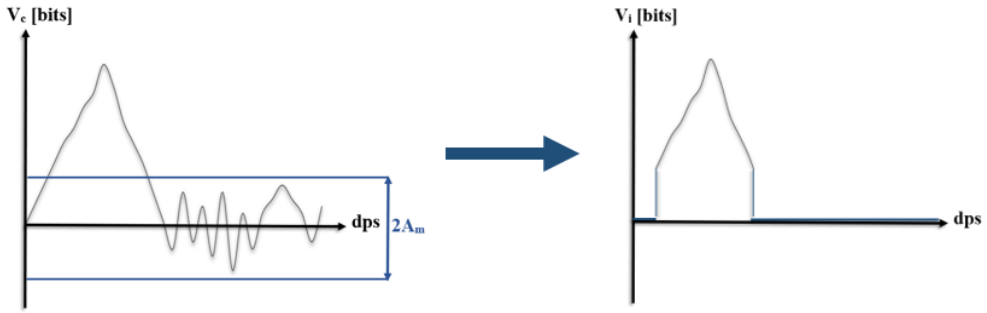


Figura 20 Ventana de medición de la señal

La aplicación del procedimiento se muestra en la Figura 21. La señal en color verde representa los valores provenientes de giroscopio. La señal en color rojo es el resultado de desplazar la señal al origen y aplicar la ventana de medición.

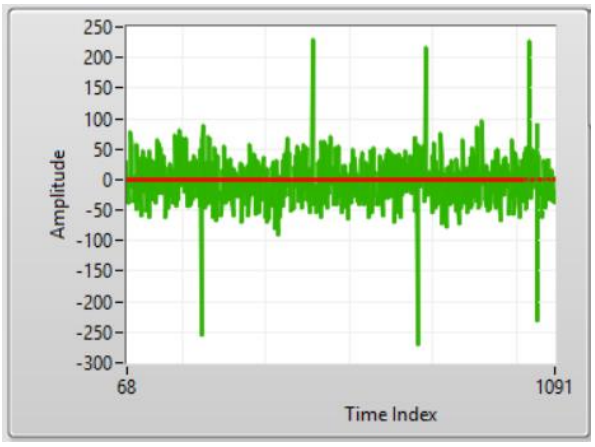


Figura 21 Resultados reales de la señal del Osciloscopio

- **Filtrado de la señal**

Este proceso se realizó mediante el filtro de Kalman bajo los lineamientos descritos por (Vera Cañal, 2015). De esta forma, el proceso del filtrado se basa en la Figura 22.

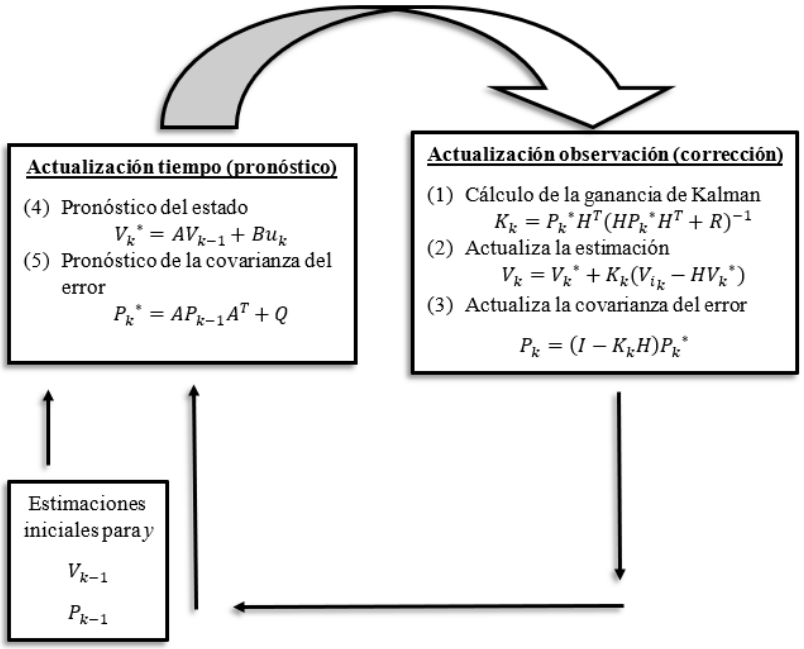


Figura 22 Proceso de filtrado de la señal

Este filtro consta de dos etapas. La etapa predictiva que calcula una estimación del estado del sistema y una estimación de la covariancia del error. La segunda es una etapa correctiva que calcula la ganancia, el estado real del sistema y la covariancia real del error.

- ***Etapa predictiva***

En esta etapa se calcula una estimación V^* del estado que se desea determinar. El estado que se busca es V que representa la velocidad angular en bits. La ecuación (3-11) rige este comportamiento, en donde el subíndice *k* indica el instante de tiempo, u es una señal de entrada y A, Bson matrices que relacionan estas variables.

$$V_k^* = AV_{k-1} + Bu_k \qquad (3\text{-}11)$$

Para el proceso se consideran perturbaciones, aceleracion angular y señal de entrada nulas entre dos instantes de tiempo. De modo que se obtiene la ecuación (3-12). Lo cual representa que se estima al sistema invariable entre dos instantes de muestreo.

$$V_k^* = V_{k-1} \qquad (3\text{-}12)$$

Se procede a estimar la covarianza del error P^* mediante la ecuación (3-13), en donde Q representa la covarianza del ruido en el proceso y P la covarianza real del error.

$$P_k^* = AP_{k-1}A^T + Q \qquad (3\text{-}13)$$

Para el valor de Q se usó 0.001 que fue determinado por (Vera Cañal, 2015). El cual presentó buenos resultados. Al remplazar el valor de Q se obtuvo la ecuación (3-14).

$$P_k^* = P_{k-1} + 0.001 \qquad (3\text{-}14)$$

- ***Etapa correctiva***

En la etapa correctiva se calcula la ganancia de Kalman, el estado actual del sistema y la covariancia del error. La ganancia del Kalman *K* se calcula mediante la ecuación (3-15) en donde *H* representa la relación entre las medidas y el estado del sistema. *R* es la covarianza del ruido en las mediciones del proceso.

$$K_k = P_k^* H^T (HP_k^* H^T + R)^{-1} \qquad (3\text{-}15)$$

Debido a que el estado es el mismo que la medición, *H* es 1. Para el valor de *R* se usó 0.07 que fue determinado por (Vera Cañal, 2015). El cual presentó buenos resultados. Al remplazar el valor de *R* y *H* se obtuvo la ecuación (3-16).

$$K_k = P_k^* (P_k^* + 0.07)^{-1} \qquad (3\text{-}16)$$

Se procedió a calcular el estado del sistema V mediante la ecuación (3-17) donde V_i es el valor medido.

$$V_k = V_k^* + K_k(V_{i_k} - HV_k^*) \qquad (3\text{-}17)$$

Debido a que H es 1, se obtiene la ecuación (3-18).

$$V_k = V_k^* + K_k(V_{i_k} - V_k^*) \qquad (3\text{-}18)$$

Finalmente se calcula el valor de la correlación del error P mediante la ecuación (3-19) en donde I es la matriz identidad.

$$P_k = (I - K_k H)P_k^* \qquad (3\text{-}19)$$

De esta forma, reemplazando I y H por 1, se obtiene la ecuación (3-20)

$$P_k = (1 - K_k)P_k^* \qquad (3\text{-}20)$$

El resultado del filtro de Kalman se muestra en la Figura 23. La señal verde corresponde a la medición V_i y la señal roja es el resultado del filtro de Kalman V.

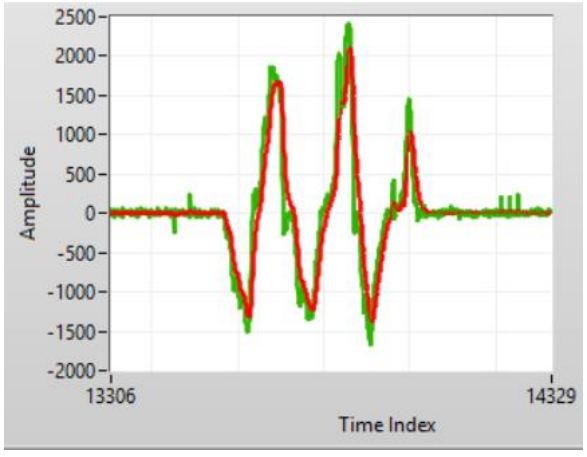


Figura 23 Resultado del Filtro de Kalman

- **Convergencia a cero**

En estado inmóvil, el filtro de Kalman produjo valores cercanos a cero. La convergencia era muy lenta y el cálculo de valores cada vez más pequeños se traducía en mayor procesamiento por parte del controlador. Por lo cual se procedió a eliminar los valores cuya magnitud sean menores a 0.01, considerándolos nulos como lo determina la ecuación (3-21)

$$V_f = \begin{cases} V, & si\ |V| > 0.01 \\ 0, & si\ |V| \leq 0.01 \end{cases} \qquad (3\text{-}21)$$

- **Integración numérica**

Una vez que se obtuvieron los valores V_f, se procedió a integrar la señal por el método del trapecio, obteniendo valores del ángulo que ha girado el robot θ_c según la ecuación (3-22).

$$\theta_{c_k} = \theta_{c_{k-1}} + \frac{V_{f_k} + V_{f_{k-1}}}{2} \cdot 0.01 \qquad (3\text{-}22)$$

- **Desbordamiento del ángulo**

La variable θ_c puede aumentar indefinidamente en caso de movimientos angulares acumulativos. A fin de evitar el desbordamiento del ángulo y debido a que las variables angulares son cíclicas, una vez que θ_c alcanza una magnitud que represente 360º se le asigna el valor de cero. En la ecuación (3-23) se muestra este procedimiento, el valor 41143 representa 360º.

$$\theta_{cf} = \begin{cases} 0, & si\ |\theta_c| > 41143 \\ \theta_c, & si\ |\theta_c| \leq 41143 \end{cases} \qquad (3\text{-}23)$$

- **Transformación**

Una vez que se obtuvo los valores del ángulo que ha girado en bits, se multiplicó por la sensibilidad del sensor (STMicroelectronics, 2010). Se obtuvo el ángulo en grados θ con la ecuación (3-24), el signo negativo corresponde a la orientación del sensor.

$$\theta = -0.00875 \cdot \theta_{cf} \qquad (3\text{-}24)$$

- **Activación**

Para evitar acumulación de errores cuando el robot está inmóvil se activa la medición mediante una variable booleana *med*. A la cual se le asigna el valor *verdadero* al iniciar una secuencia de movimiento.

D. Movimiento angular

El movimiento angular se controla mediante una función llamada *giro*. La cual se muestra en la Figura 24 que recibe como parámetros la magnitud, dirección y sentido del movimiento lineal a realizar y el ángulo actual del robot. Las salidas de la función son el ángulo deseado, la velocidad y el sentido del giro y si se debe continuar con el movimiento angular. Las variables de entrada y salida de la función *giro* se muestran en la Tabla 18.

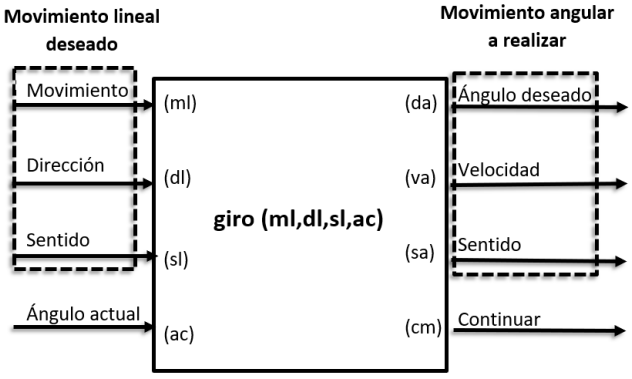


Figura 24 Representación del Movimiento angular

Tabla 18

Entradas y salidas de la función Giro

Variable	Tipo	Característica
Magnitud del movimiento lineal (ml)	Real	Dato numérico correspondiente al desplazamiento lineal a realizar
Dirección del movimiento lineal (dl)	Booleano	Verdadero si el movimiento es en el eje X, Falso si el movimiento es en el eje Y
Sentido del movimiento lineal (sl)	Booleano	Verdadero si el movimiento es hacia atrás. Falso si el movimiento es hacia adelante.
Ángulo actual (ac)	Real	Dato numérico correspondiente a la orientación del robot
Angulo deseado (ad)	Real	Ángulo al que debe llegar el robot
Velocidad del movimiento angular (va)	Real	Dos tipos de velocidades 0.7 para un movimiento angular rápido, 0.4 para un movimiento angular lento.
Sentido del movimiento angular (sa)	Booleano	Verdadero si el giro debe ser antihorario. Falso si el giro debe ser horario.
Continuar el movimiento angular (cm)	Booleano	Verdadero si se debe continuar el movimiento angular, de lo contrario, falso.

Para determinar cada una de las salidas de la función giro se hace uso de uno o más datos de entrada. El cálculo de la magnitud del movimiento angular sigue la lógica mostrada en la Figura 25. Se evalúa la dirección del movimiento lineal, si es en el eje X el ángulo deseado puede ser 0º o 180º. Si la magnitud del movimiento lineal es positiva, el ángulo deseado es 0º, de lo contrario es 180º. Si el sentido del movimiento lineal es hacia atrás, se invierten las asignaciones. Si el movimiento lineal es en el eje Y, se sigue la misma lógica, pero el ángulo deseado puede ser -90º o 90º.

```
1  ad = giro(dl,sl,ml)
2  inicio
3  Si (dl == verdadero) entonces:
4    Si ((ml >= 0) xor sl) entonces:
5      ad = 0
6    Si no:
7      ad =  180
8  Si no:
9    Si ((ml >= 0) xor sl) entonces:
10     ad = 90
11   Si no:
12     ad =  -90
13 fin
```

Figura 25 Pseudocódigo para calcular el ángulo deseado

La velocidad de giro se determina siguiendo la lógica mostrada en la Figura 26. Se inicia calculando el error que representa el desplazamiento angular que se debe realizar. Si el error es mayor a 10º la velocidad angular es 0.7 correspondiente a un movimiento rápido de acercamiento. Si el error se encuentra entre 1º y 10º la velocidad angular es 0.4 correspondiente a un movimiento lento de precisión, el cual se mantiene durante 50 milisegundos para permitir una nueva evaluación. Si el error es menor a 1º, el robot se detiene.

```
va = giro(ac,ma)
inicio
error = |ma-ac|
Si (error > 10)entonces:
  va = 0.7
Si no:
  Si (error > 1) entonces:
    va = 0.4
    esperar 50 milisegundos
    va = 0
  Si no:
    va = 0
fin
```

Figura 26 Pseudocódigo para calcular la velocidad de giro

El sentido de giro se determina siguiendo la lógica mostrada en la Figura 27. Se evalúa si ha existido un cambio de signo entre el ángulo deseado y el ángulo actual mediante la multiplicación entre ambos. Si no ha existe un cambio de signo se gira siguiendo el sentido del error, si es positivo *sa* es falso, correspondiente a un giro horario, de lo contrario *sa* es verdadero, correspondiente a un giro antihorario. Si ha existido un cambio de signo se evalúa el ángulo deseado. Si el ángulo deseado es mayor a cero *sa* es verdadero, si es menor a cero *sa* es falso, si el ángulo deseado es cero se evalúa el ángulo actual. Si el ángulo actual es mayor o igual a cero *sa* es falso, de lo contrario *sa* es verdadero.

```
sa = giro(ad,ac)
inicio
aux = ad*ac
error = ad-ac
Si (aux > 0)entonces:
  Si (error >= 0) entonces:
    sa = falso
  Si no:
    sa = verdadero
Si no:
  Si (ad > 0) entonces:
    sa = verdadero
  Si (ad < 0) entoces:
    sa = falso
  Si no:
    Si (ac >= 0) entonces:
      sa =  falso
    Si no:
      sa = verdadero
fin
```

Figura 27 Pseudocódigo para determinar el sentido de giro

La continuación del movimiento angular se determina siguiendo la lógica mostrada en la Figura 28. Se calcula la magnitud del error, si es mayor a 1º el movimiento debe continuar, de lo contrario, cm es falso.

```
1  cm = giro(ad,ac)
2  inicio
3  error = |ad-ac|
4  Si (error > 1) entonces:
5    cm = verdadero
6  Si no:
7    cm = false
8  fin
```

Figura 28 Pseudocódigo para determinar la continuación del movimiento angular

El proceso de llegar a la orientación deseada sigue la lógica mostrada en la Figura 29. Se evalúa si se debe continuar el movimiento y se siguen las secuencias mientras cm sea verdadero. La pausa de 100 milisegundos permite al controlador atender otras tareas.

```
1  inicio
2  cm = giro(ad,ac)
3  mientras (cm == verdadero) hacer:
4    giro(ml,dl,sl,ac);
5    esperar 100 milisegundos
6  fin mientras
7  fin
```

Figura 29 Pseudocódigo para llegar a la orientación deseada

E. Movimiento lineal

El movimiento lineal se controla mediante una función llamada *mlineal*. La cual se muestra en la Figura 30. La función recibe como parámetros la magnitud, sentido y velocidad del movimiento, el ángulo actual y deseado, la velocidad angular, la distancia de acercamiento y una marca de pausa. Las salidas de la función son la velocidad y sentido de giro de los motores, una señal de alarma y si se debe continuar con el movimiento lineal. Las características de entradas y salidas de la función *mlineal* se muestran en la Tabla 19.

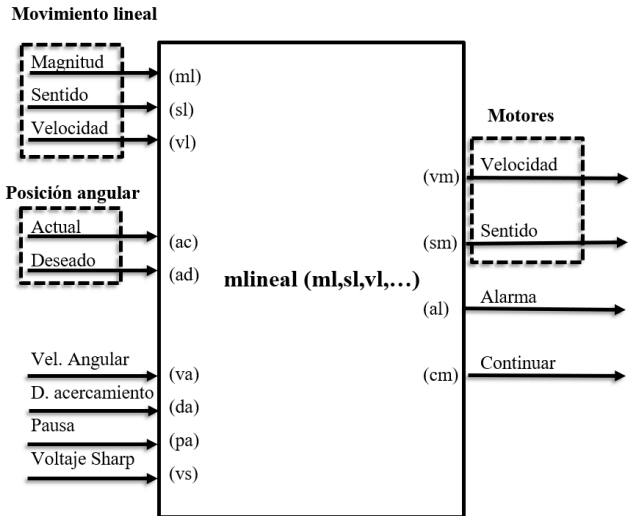


Figura 30 Representación del Movimiento lineal

Tabla 19

Entradas y salidas de la función mlineal

Variable	Tipo	Característica
Magnitud del movimiento lineal (ml)	Real	Dato numérico correspondiente a la distancia entre el punto actual y el deseado.
Sentido del movimiento lineal (sl)	Booleano	Verdadero si el movimiento es hacia delante. Falso si el movimiento es hacia atrás
Velocidad del movimiento lineal (vl)	Real	Velocidad deseada para el movimiento
Posición angular actual (ac)	Real	Dato numérico correspondiente a la posición angular actual.
Posición angular deseada (ad)	Real	Dato numérico correspondiente a la posición angular deseada.
Velocidad angular (va)	Real	Dato numérico correspondiente a la velocidad angular medida.
Distancia de acercamiento (da)	Real	Dato numérico correspondiente a la distancia de acercamiento al punto deseado.
Pausa (pa)	Booleano	Verdadero si se desea detener el movimiento momentáneamente.
Voltaje del sensor sharp IR (vs)	Real	Voltaje proveniente del sensor Sharp IR
Velocidad de los motores (vm)	Real	Dato numérico correspondiente a la velocidad de los motores
Sentido de los motores (sm)	Booleano	Verdadero si el sentido es horario. Falso si el sentido es antihorario.
Alarma (al)	Booleano	Verdadero si existe una alarma
Continuar (cm)	Booleano	Verdadero si se debe continuar con el movimiento lineal.

Para determinar cada una de las salidas de la función *mlineal* se hace uso de uno o más datos de entrada. El cálculo de la velocidad de los motores sigue la lógica mostrada en la Figura 31. Si la distancia actual al punto deseado es mayor a distancia de acercamiento, *pa* es falso y el voltaje del Sharp *vs* es menor a 2.5 [V], valor que corresponde a 9 centímetros de distancia, *vm* es igual a la velocidad *vl*, de lo contrario, *vm* es cero.

```
1  vm = mlineal(ml,vl,da,pa,vs)
2  inicio
3  Si (ml>da y pa == falso y vs<2.5) entonces:
4    vm = vl
5  Si no:
6    vm = 0
7  fin
```

Figura 31 Pseudocódigo para calcular la velocidad de los motores

El sentido de giro de los motores se determina siguiendo la lógica mostrada en la Figura 32. Si el sentido del movimiento es verdadero correspondiente a un movimiento hacia adelante, el sentido de giro es verdadero, correspondiente a un giro horario de los motores.

```
1  sm = mlineal (sl)
2  inicio
3  Si (sl == verdadero) entonces:
4    sm = verdadero
5  Si no:
6    sm = falso
7  fin
```

Figura 32 Pseudocódigo para determinar el sentido de giro de los motores

La alarma se produce siguiendo la lógica en la Figura 33. Si el voltaje del Sharp IR *vs* es mayor a 2.5 [V] la alarma *al* es verdadero, de lo contrario es falso. El valor 2.5 [V] corresponde a una distancia aproximada de 9 [cm], lo que indica un obstáculo frontal en el robot.

```
1  al = mlineal(vs)
2  inicio
3  Si (vs>2.5) entonces:
4    al = verdadero
5  Si no:
6    al = falso
7  fin
```

Figura 33 Pseudocódigo para especificar los casos de alarma

La continuación del movimiento lineal se determina siguiendo la lógica mostrada en la Figura 34. Se calcula la magnitud del error en la posición angular. Si el error es menor a 3º y la distancia entre el punto actual y el deseado es mayor a la distancia de acercamiento, entonces *cm* es verdadero, de lo contrario es falso.

```
1  cm = mlineal(ml,da,ac,ad)
2  inicio
3  error = |ad-ac|
4  Si (ml>da y error<3)
5    cm = verdadero
6  Si no:
7    cm = falso
8  fin
```

Figura 34 Pseudocódigo para determinar el movimiento lineal

El proceso de llegar a la orientación deseada sigue la lógica mostrada en la Figura 35 Se evalúa si se debe continuar el movimiento y se siguen las secuencias mientras *cm* sea verdadero. La pausa de 100 milisegundos permite al controlador atender otras tareas.

```
1  inicio
2  cm = mlineal(ml,da,ac,ad)
3  mientras (cm == verdadero) hacer:
4    mlineal(ml,sl,vl,ac,ad,va,da,pa,vs)
5    esperar 100 milisegundos
6  fin mientras
7  fin
```

Figura 35 Pseudocódigo para llegar a la orientación deseada

F. Corrección de trayectoria

La corrección de la trayectoria permite que el robot mantenga su orientación mientras se desplaza. La función *mlineal* realiza la acción de control *x* mediante la ecuación (3-25). En donde el subíndice *k* es el instante de tiempo, *va* es la velocidad angular medida.

$$x_k = x_{k-1} - 0.02 \cdot va \qquad (3\text{-}25)$$

El valor 0.02 afecta a la velocidad de corrección y se determinó experimentalmente. La acción de control se limita entre -0.1 y 0.1 mediante la ecuación (3-26), estos límites corresponden a un giro positivo y negativo del robot.

$$x_{kf} = \begin{cases} x_k, & si\ |x_k| \leq 0.1 \\ 0.1, & si\ x_k > 0.1 \\ -0.1, & si\ x_k < -0.1 \end{cases} \qquad (3\text{-}26)$$

La velocidad del motor de la rueda derecha se mantiene constante mientras que la velocidad de la rueda izquierda se modifica según la ecuación (3-27).

$$vm_i = vm + x_{kf} \qquad (3\text{-}27)$$

G. Secuencia de movimiento

La ubicación del robot desde un punto inicial hacia un punto deseado hace uso de las funciones *giro* y *mlineal* siguiendo la lógica mostrada en la Figura 36. El primer ciclo *mientras* evalúa la variable *ir* cada 200 milisegundos, una vez que *ir* es verdadero se calcula la distancia del centro al eje X *dx*, al eje Y *dy* y la distancia total *d*. El ciclo *mientras* se ejecuta cuando la distancia *d* es mayor a 3 y si la variable *cd* es verdadero, lo cual que permite activar la corrección por distancia total.

```
inicio
mientras (ir == falso) hacer:
esperar 200 milisegundos
fin mientras
medir = verdadero
dx = |xd-xc|
dy = |yd-yc|
d = raiz(dx^2+dy^2)
mientras (d>3 y cd == verdadero) hacer:
  dx = |xd-xc|
  mientras (dx>2) hacer:
    giro(dx,verdadero,falso,ac)
    mlineal(dx,verdadero,0.85,ac,ad,va,20,pa,vs)
    mlineal(dx,verdadero,0.60,ac,ad,va,2,pa,vs)
  fin mientras
  dy = |yd-yc|
  mientras (dy>2) hacer:
    giro(dy,falso,falso,ac)
    mlineal(dy,falso,0.85,ac,ad,va,20,pa,vs)
    mlineal(dy,falso,0.60,ac,ad,va,2,pa,vs)
  fin mientras
fin mientras
fin
```

Figura 36 Pseudocódigo para determinar la secuencia de movimiento

El primer ciclo realiza el movimiento en el eje X mientras *dx* sea mayor a 2. Se ejecuta la función *giro* con los datos necesarios para este movimiento, posteriormente se usa la función *mlineal* con los parámetros para un movimiento rápido de acercamiento seguida por la misma función con parámetros para un movimiento lento de precisión. Una vez concluido el movimiento en el eje X se procede a realizar el movimiento en el eje Y con la misma lógica.

H. Actuador

El actuador es un gripper con un grado de libertad (Pitch) que permite inclinar la pinza para recoger piezas del piso. La apertura y la inclinación del actuador se controlan mediante servomotores MG955 (Jones). El controlador envía pulsos PWM a una frecuencia de 50 Hz. La apertura y la inclinación se manipulan mediante porcentaje de apertura A_p y porcentaje de inclinación I_p. El ciclo de trabajo ct para los pulsos PWM se calculan mediante la ecuación (3-28), En donde X_p se reemplaza por A_p o I_p según corresponda.

$$ct = \frac{X_p \cdot 0.034 + 4}{100} \qquad (3\text{-}28)$$

3.3. Subsistema Arquitectura IoT

3.3.1. Requerimientos

La arquitectura debe cumplir los criterios necesarios para ser considerada IoT además de maximizar el uso de los recursos disponibles en el Laboratorio de Mecatrónica y Sistemas Dinámicos de la universidad. Los criterios IoT que se consideró fueron los definidos por (Mahmoud, Yousuf, Aloul, & Zualkernan, 2016) , que consisten en tecnología distribuida, interacción de objetos conectados, seguridad, escalabilidad y eficiencia Energética. La importancia de las características se evaluó mediante matrices de selección y su resultado se muestra en la Tabla 20.

Tabla 20
Importancia de los requerimientos de la arquitectura IoT

Característica	**Importancia [%]**
Tecnología distribuida	14
Escalabilidad	17
Eficiencia energética	5
Interacción	19
Seguridad	17
Disponibilidad de recursos en el laboratorio	29

3.3.2. Diseño

El diseño de la Arquitectura IoT consistió en la elección e implementación de sus componentes principales, los cuales son:

- Red
- Protocolo
- Tipo de servicio

A. Proceso de elección

La elección para cada una de las opciones se evalúa mediante matrices de selección en función a su cumplimiento de los requerimientos definidos en la Tabla 20. Los resultados para la elección de la red se muestran en la Tabla 21.

Tabla 21
Resultados de la elección de la red

Red	Calificación
WiFi	34.29%
Zigbee	14.05%
6LoWPAN	31.43%
RFID	20.24%

La red que se eligió fue WiFi. Una vez determinada la red se procedió a la elección del protocolo entre cuatro opciones capaces de trabajar en WiFi. Los resultados para la elección del protocolo se muestran en la Tabla 22.

Tabla 22
Resultados de la elección del protocolo

Protocolo	Calificación
MQTT	28.81%
HTTP-REST API	18.57%
CoAP	26.31%
MQTT-SN	26.31%

El protocolo que se eligió fue MQTT. Una vez determinado el protocolo se procedió a la elección del tipo de servicio. Los resultados se muestran en la Tabla 23.

Tabla 23
Ponderaciones para la elección del tipo de servicio

Modelo	Calificación
IaaS	38.89%
PaaS	36.11%
SaaS	25.00%

El tipo de servicio que se eligió fue IaaS. Una vez determinado el tipo de servicio se procedió a la elección del proveedor. Las empresas que van a la vanguardia en la infraestructura como servicio son:

- Digital Ocean
- Linode
- Vultr

Los tres proveedores ofrecen diferentes planes mensuales en los que el usuario puede tener su propia cuenta y crear una máquina virtual. La comparación de las características para un servicio básico se muestra en la Tabla 24.

Tabla 24

Características de los proveedores de Infraestructuras como servicio

	Memoria	**vCPUs**	**Disco ssd**	**Transferir**	**Precio**
Digital Ocean	1 GB	1 vCPU	25 GB	1 TB	$ 5 / mes
Linode	1 GB	1 vCPU	20 GB	1 TB	$ 5 / mes
Vultr	1 GB	1 vCPU	25 GB	1 TB	$ 5 / mes

Una ligera ventaja en el espacio de disco coloca a Digital Ocean y Vultr como opciones principales. Digital Ocean proporciona un crédito gratuito de $100 para probar el servicio, por lo cual se eligió a este proveedor.

B. Arquitectura IoT

La implementación de la arquitectura IoT para robótica colaborativa se conforma de cinco elementos que se muestran en la Figura 37, donde A es el bróker MQTT, B es el sistema Multi-Robot, C es la interfaz de usuario, D es el planificador y E son los dispositivos desde los que accede el usuario.

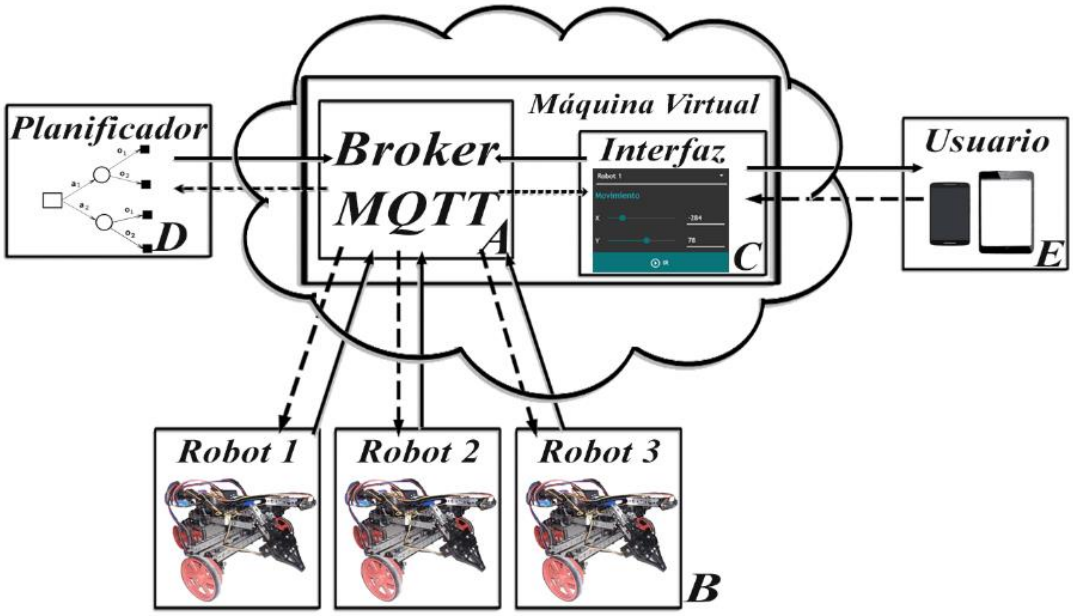


Figura 37 Arquitectura IoT basada en MQTT para robótica colaborativa. A: Broker MQTT. B: Sistema Multi-Robot. C: Interfaz de usuario. D: Planificador. E: Usuario

- ***Interconexión***

La interconexión se realiza mediante el protocolo MQTT. Los dispositivos generadores de información publican en el bróker MQTT bajo un tópico representativo al cual, los dispositivos consumidores de información están suscritos. (Collina, Corazza, & Vanelli-Coralli, 2012).

Se desarrolló una estructura para los tópicos que facilite la construcción y comprensión de los mismos, de forma que constan de tres partes *A/B/C*, en donde *A* corresponde al nombre del dispositivo de la red que publica en el bróker, pudiendo ser tres: *rbx* para cualquier robot x de la red, *pl* para el planificador e *it* para la interfaz. *B* es el nombre de la variable a publicar y *C* es una subvariable en caso de ser necesario. En la Tabla 25 se resumen la construcción de los tópicos principales en la interconexión.

Tabla 25

Construcción de los tópicos principales

A[a]	B[a]	C[a]	Tópico[a]	Suscriptor
Robot x (rbx[b])	Posición del centro: xc, yc	-	rbx/xc rbx/yc	Interfaz
	En movimiento o parado: mv	-	rbx/mv	Interfaz
	Alarma: al	-	rbx/al	Interfaz
Interfaz (it)	Posición deseada: xd,yd	-	it/xd it/yd	Planificador
	Apertura e inclinación del actuador: ap, ic	-	it/ap it/ic	Planificador
	Orden de mover: ir	-	it/ir	Planificador
	Robot a controlar: rb	Número de robot: x	it/rb/x	Planificador
Planificador (pl)	Robot a manipular: robot (rb) + número (x)	Posición deseada: xd, yd	pl/rbx/xd	Robot 1
		Apertura e inclinación del actuador: ap, ic	pl/rbx/ap pl/rbx/ic	Robot 1
		Orden de mover: ir	pl/rbx/ir	Robot 1

[a.] A corresponde al dispositivo publicador, B es la variable a publicar y C es una subvariable, en caso de existir. El tópico se construye de forma: A/B/C.

[b.] rbx representa a los robots rb1, rb2 y rb3

- ***Broker Mqtt***

El bróker MQTT formula como intermediario en la comunicación. Los dispositivos conectados lo hacen mediante la dirección del bróker, eliminando la necesidad de especificar la dirección del miembro de la red que requiere la información. Los datos provenientes de los dispositivos de la red se publican en un tópico representativo en el bróker, quedando a disposición de los miembros que se suscriban al mismo tópico.

- ***Robots***

Mediante la conexión Wifi que dispone la tarjeta NI MyRIO se conectó al bróker haciendo uso de una librería para MQTT (Cowen, 2018) a la cual se modificó para permitir a los robots móviles:

a. Conectarse al bróker MQTT y suscribirse a los tópicos detallados en la Tabla 26.

b. Almacenar los mensajes recibidos en variables internas. Una vez que se recibe un mensaje, se asigna su valor a una variable cuyo nombre es la parte final del tópico, así, la posición deseada para cualquier robot x publicada bajo el tópico *pl/rbx/xd* se almacena en el robot x en una variable llamada *xd*, como se muestra en la Tabla 26.

c. Publicación automática de su posición, estado (en movimiento o parado) y alarma bajo los tópicos mostrados en la Tabla 27

Tabla 26

Tópicos a los que se suscriben los robots

Tópico	**Variable interna**	**Significado**
pl/rbx/xd[a]	xd	Posición deseada en x
pl/rbx/yd	yd	Posición deseada en y
pl/rbx/ap	ap	Apertura del actuador
pl/rbx/ic	ic	Inclinación del actuador
pl/rbx/ir	ir	Orden de moverse
pl/rbx/pa	pa	Pausar el movimiento
pl/rbx/cl	cl	Color del objetivo asignado
pl/rbx/lb	lb	Iniciar secuencia de movimiento automático
pl/rbx/enp	enp	Orden de publicar en el tópico rbx/enr, el mensaje “1”.

[a]. rbx representa a los robots rb1, rb2 y rb3

Tabla 27

Tópicos en los que publican los robots pie de tabla

Tópico	Característica
/rbx/xc[a]	Posición x del centro del robot
/rbx/ yc	Posición y del centro del robot
/rbx/ mv	Estado del robot: En movimiento o parado
/rbx/ al	Alarma en caso de obstáculo frontal
/rbx/ di	Disponibilidad del robot para una nueva tarea
/rbx/ es	Estado del robot: Dentro o fuera del espacio de trabajo
/rbx/ enr	Respuesta del robot para verificar su encendido

[a]. rbx representa a los robots rb1, rb2 y rb3.

- ***Interfaz***

- **Funciones**

La interfaz se desarrolló mediante la herramienta Node-RED (Blackstock & Lea, 2015) y cumple las siguientes funciones:

a. Comunicarse con el bróker MQTT y presentar los datos de interés bajo el protocolo http, lo que garantiza su acceso desde cualquier navegador web.
b. Proveer una interfaz de control que permita al usuario manipular a los robots.
c. Permitir la escalabilidad y la adición de servicios de forma simple.

- **Partes y comunicación**

La interfaz de usuario consta de una ventana de control y una ventana de monitoreo. La comunicación con la Arquitectura se realiza mediante el bróker MQTT.

a. Ventana de control. - Esta ventana es la encargada de publicar las ordenes de control en el bróker MQTT, cuenta con 5 zonas como se muestra en la Figura 38. La apariencia real de la ventana se muestra en la Figura 39, los elementos que la conforman se resumen en la Tabla 28 y los tópicos sobre los que publica se muestran en la Tabla 29.

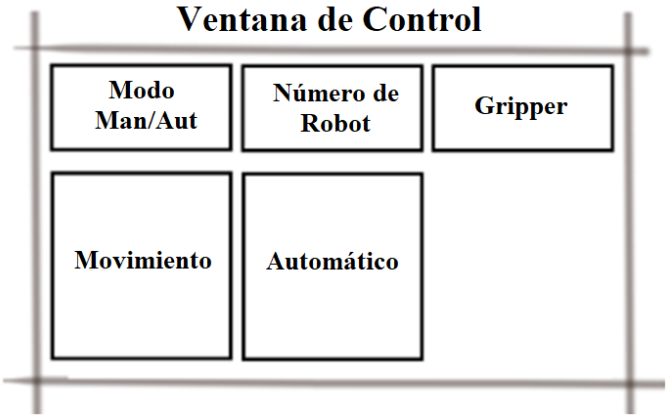


Figura 38 Distribución de la ventana de control

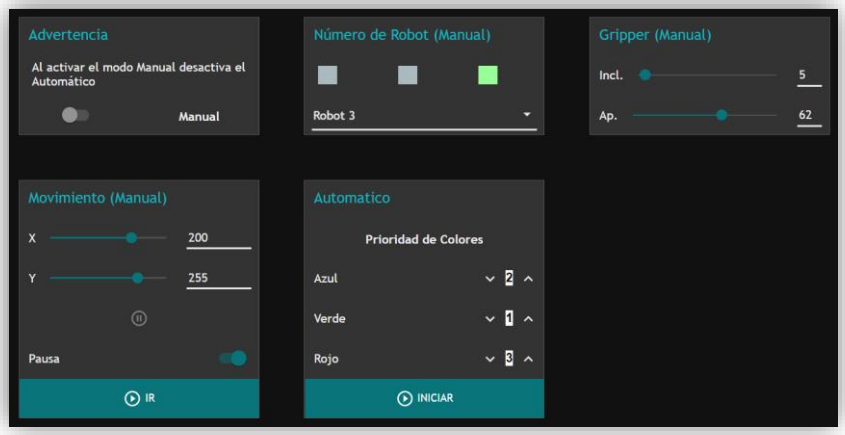


Figura 39 Apariencia real de la ventana de control

Tabla 28

Descripción de las partes de la ventana de control

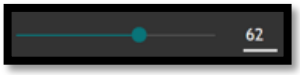
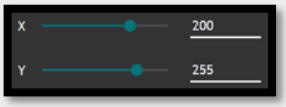
Objeto	Descripción	Funcionamiento
	Switch M/A	Permite seleccionar el modo de operación: Manual/ Automático.
	Selector robot	Permite escoger el robot a manipular.
	Control Inclinación Grippper	Consiste en dos controles numéricos que permiten manipular la inclinación y apertura del gripper de los robots.
	Control Apertura Grippper	
	Control de Movimiento (X, Y)	Consiste en dos controles numéricos que permiten seleccionar el punto al cual se moverá el robot, un botón de partida y uno de pausa.
	Interruptor de pausa	
	Botón Ir	
	Selector prioridad de color	Contiene un selector de prioridades del color de los objetivos a mover y un botón de iniciar el modo automático.
	Botón Iniciar automático	

Tabla 29: ***Tópicos en los que publica la interfaz***

Tópico	Característica
it/md	Modo manual o automático
it/zu	Número de robot a manipular
it/ap	Apertura del actuador
it/ic	Inclinación del actuador
it/xd	Posición deseada en x
it/yd	Posición deseada en y
it/ir	Orden de moverse
it/pa	Pausar el movimiento
it/pr/az	Prioridad de los objetivos azules
it/pr/vd/	Prioridad de los objetivos verdes
it/pr/rj	Prioridad de los objetivos rojos
it/in	Orden de iniciar el modo automático

b. Ventana de monitoreo. - Esta ventana es la encargada de suscribirse a los tópicos importantes en el bróker MQTT, cuenta con 3 zonas como se muestra la Figura 40, los elementos que la conforman se resumen en la Tabla 30 y los tópicos a los que se suscribe se muestran en la Tabla 31.

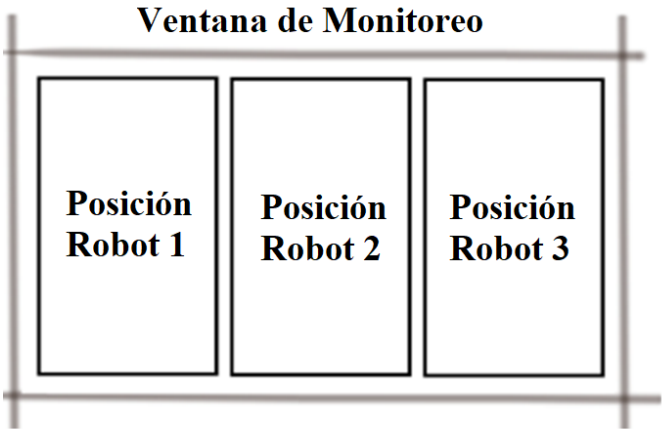


Figura 40 Distribución de la ventana de monitoreo

Figura 41 Apariencia real de la ventana de monitoreo

Tabla 30

Descripción de las partes de la ventana de control

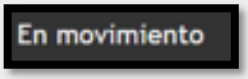
Objeto	Descripción	Funcionamiento
	Indicador de estado: Encendido y apagado	Permite monitorear la posición, estado y alarmas de los robots 1, 2 y 3.
	Indicador de posición	
	Indicador de estado: En movimiento y parado.	

Tabla 31

Tópicos a los que se suscribe la interfaz

Tópico	Característica
rbx/xc[a]	Posición x del centro del robot
rbx/ yc	Posición y del centro del robot
rbx/ mv	Estado del robot: En movimiento o parado
rbx/ al	Alarma en caso de obstáculo frontal
pl/rbx/ en	Robot encendido

a. rbx representa a los robots rb1, rb2 y rb3

- ***Planificador***

El planificador cumple tres funciones. Verificar los robots encendidos del sistema, canalizar los datos provenientes desde la interfaz de usuario hacia los robots y ejecutar el control colaborativo. Su inclusión permitió la integración del sistema por lo cual se detalla su desarrollo en una sección individual.

3.4. Integración del Sistema

La integración del sistema se realizó mediante la inclusión de un planificador automático encargado de comunicar, centralizar y controlar al sistema Multi-Robot. Los parámetros correspondientes al entorno y la dinámica colaborativa influyen en la complejidad del planificador. Por lo cual, se procedió a definir la aplicación colaborativa y el entorno de trabajo para el sistema Multi-Robot.

3.4.1. Aplicación

La aplicación consiste en llevar al entorno de un estado inicial hacia un estado final como se ilustra en la Figura 42. En el estado inicial un grupo de piezas, llamadas objetivos, están distribuidas en una zona de trabajo y los robots se encuentran en puntos específicos de partida. El estado final es la zona de trabajo libre de objetivos, los cuales deben encontrarse en una zona determinada y los robots deben haber vuelto a sus puntos iniciales. Para llegar al estado final, los objetivos deben ser trasladados en función de su color. El usuario determina la prioridad con la que se debe organizar las piezas. En la Figura 43 se muestran los estados en el entorno real de trabajo.

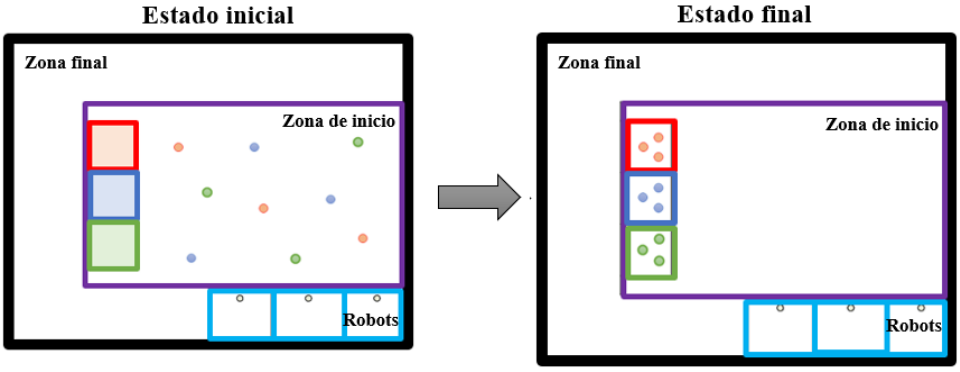


Figura 42 Esquema entre el estado inicial y final de la aplicación

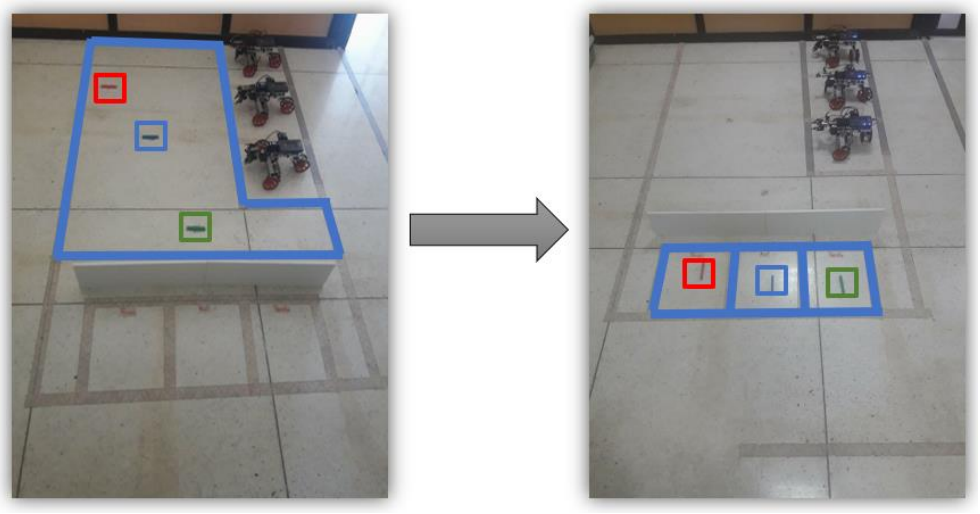


Figura 43 Cambio entre el estado inicial y final de la aplicación en el entorno real

3.4.2. Entorno de trabajo

En función de la aplicación colaborativa y al espacio provisto por parte del Laboratorio de Mecatrónica y Sistemas Dinámicos de la universidad, se definió el entorno de trabajo con las dimensiones que se muestran en la Figura 44.

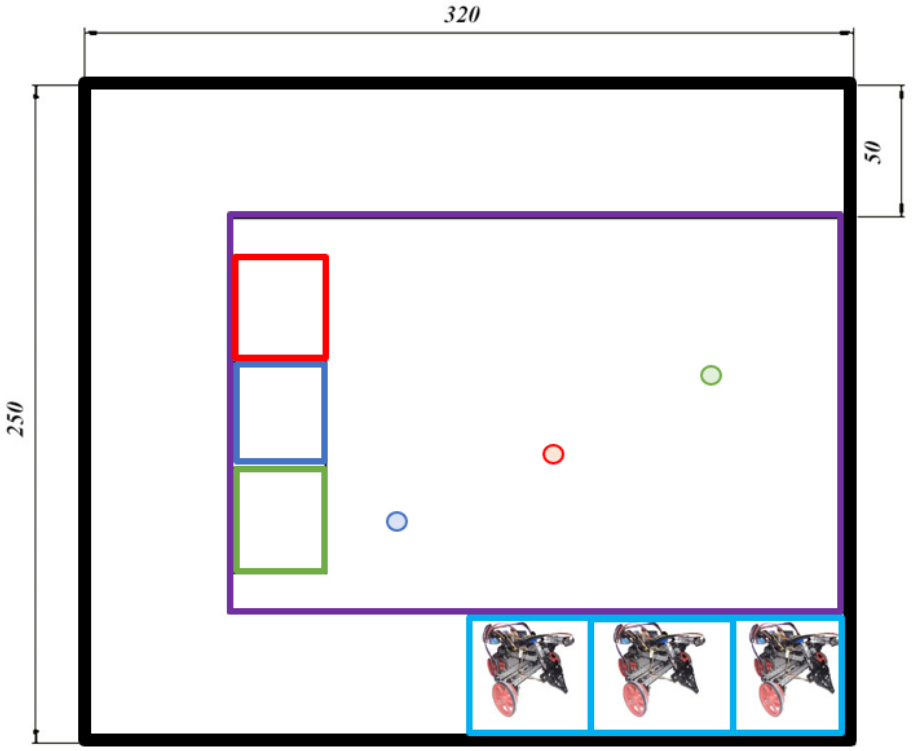


Figura 44 Entorno de trabajo

El entorno de trabajo dispone de diferentes zonas con propósitos específicos. La zona inicial que se muestra en la Figura 45 es el lugar donde los robots esperan las ordenes provenientes del planificador o el usuario. Cada división esta asignada a un robot específico. La programación del robot hace uso de dicho punto para adecuar su sistema de navegación.

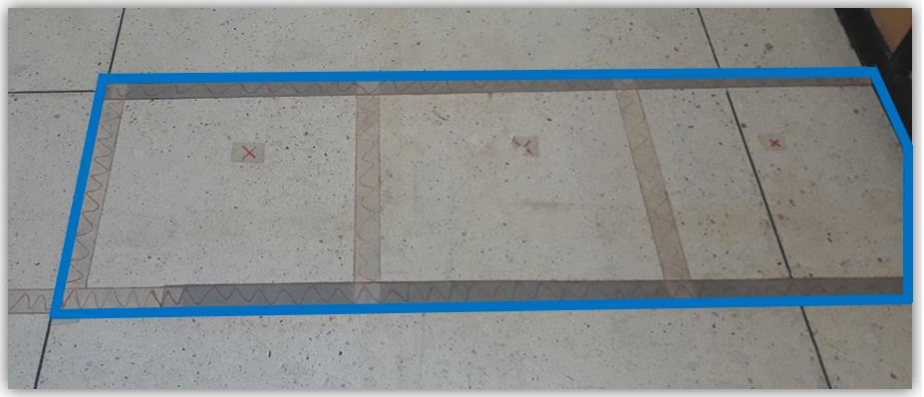


Figura 45 Zona inicial

La zona de trabajo se muestra en la Figura 46. La cual es el área donde estarán dispuestos los objetivos en el estado inicial del entorno. Debido a su tamaño fue necesario limitar a uno el número de robots simultáneos que pueden estar dentro de la zona.

Figura 46 Zona de trabajo

La zona de final o de desecho se muestra en la Figura 47. La cual está dividida en tres áreas, cada una corresponde al destino final de los objetivos dependiendo de su color.

Figura 47 Zona final

La zona de movilización se muestra en la Figura 48. La cual tiene una forma de *C* que rodea a las demás zonas, su función es permitir a los robots un libre tránsito y evitar colisiones mientras están manipulando los objetivos.

Figura 48 Zona de tránsito

3.4.3. Planificador automático

El planificador automático es un algoritmo de control y comunicación que se ejecuta en un computador conectado a la red IoT mediante el bróker MQTT. Las funciones principales del planificador son verificar el número de robots disponibles, canalizar la información desde la interfaz a los robots y realizar el control colaborativo para el sistema Multi-Robot.

La verificación de los robots encendidos se realiza mediante una comunicación constante con los mismos. El planificador inicia su secuencia declarando un contador *i* que representa el número de veces que se no se ha recibido respuesta del robot, Figura 49. El planificador publica sobre el

tópico *pl/rbx/enp* el mensaje "1", se suscribe al tópico *rbx/enr* y almacena los mensajes en la variable *enr*. Si se ha recibido un "1" en *enr*, el robot esta encendido, se inicializa el contador y se vuelve a evaluar después de 200 milisegundos, de no ser así, se espera 300 milisegundos y se aumenta el contador. Si el contador es menor a 15 se asume el robot como encendido, de lo contrario, se asume como apagado. En todos los casos, se repite el proceso. El robot inicia su secuencia suscribiéndose al tópico *pl/rbx/enp* y almacena los mensajes en la variable *enp*, Figura 50. Si *enp* es "1", el robot publica en *rbx/enr* el mensaje "1". El proceso se repite después de la evaluación.

```
1  Inicio
2  i = 0
3  mientras (verdadero)
4    Publicar 1 en pl/rbx/enp
5    Suscribirse en rbx/enr
6    enr = mensajemqtt
7    Si (enr == 1) entonces:
8      i = 0
9      Robot x encendido
10     Esperar 200 milisegundos
11   Si no:
12     Esperar 300 milisegundos
13     i = i+1
14     Si (i>15) entonces:
15       Robot x apagado
16     Si no:
17       Robot x encendido
18 fin mientras
19 fin
```

Figura 49 Verificación de encendido, secuencia que ejecuta el planificador. rbx representa a los robots rb1, rb2 y rb3

```
1  inicio
2  mientras (verdadero)
3    Suscribirse a pl/rbx/enp
4    enp = mensajeqtt
5    Si (enp == 1) entonces:
6      Publicar 1 en rbx/enr
7  fin mientras
8  fin
```

Figura 50 Verificación de encendido, secuencia que ejecuta el robot. rbx representa a los robots rb1, rb2 y rb3.

Para la canalización de los datos provenientes desde la interfaz de usuario hacia los robots se incluyó un selector de la unidad que se desea manipular. Con lo cual se evitó aumentar cinco controles por cada robot en la arquitectura. Se dispone de una variable numérica para la posición deseada en *x* y *y*, el porcentaje de apertura, la inclinación del actuador, y un botón para dar la orden de movimiento.

El planificador realiza esta canalización mediante la creación dinámica de tópicos en función del robot seleccionado. En la Figura 51 se ejemplifica este proceso mediante el movimiento y monitoreo de cualquier robot x a un punto deseado.

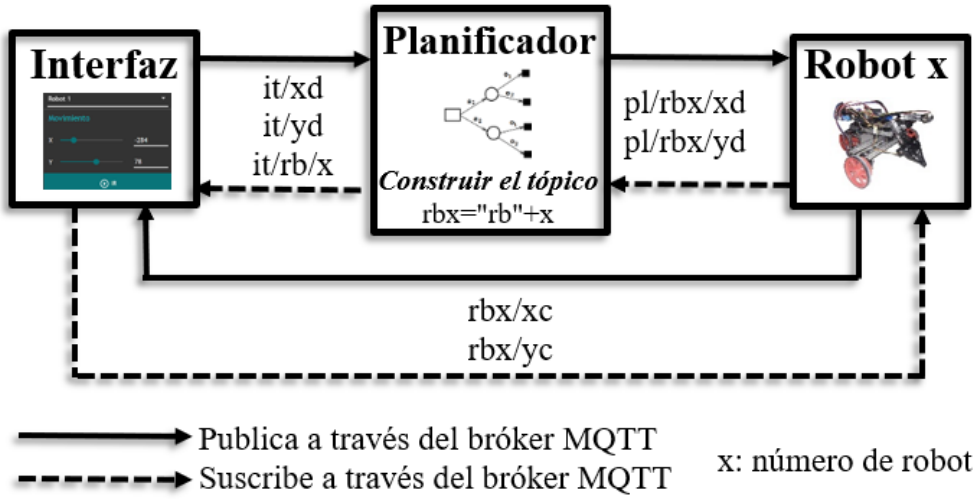


Figura 51 Ejemplo de comunicación para el movimiento y monitoreo de cualquier robot x a un punto deseado.

La interfaz publica las variables y el número *x* del robot a controlar en los tópicos *it/xd*, *it/yd* y *it/rb/x* a los cuales el planificador está suscrito. El planificador construye dinámicamente nuevos tópicos con el número de robot definido por la interfaz, posteriormente publica sobre *pl/rbx/xd* y *pl/rbx/yd* las coordenadas del punto deseado. Finalmente, el robot suscrito a estos tópicos realiza su movimiento y publica los cambios en *rbx/xc* y *rbx/yc* a los cuales está suscrita la interfaz.

El control colaborativo empieza cuanto el usuario ha elegido la prioridad para la organización de los objetivos y envía la orden de iniciar. La ubicación y color de las piezas se asumen como datos de entrada para el planificador. Los cuales consisten en una matriz, la ubicación en *x*, *y* y el color de los objetivos se agrupan en la primera, segunda y tercera columna respectivamente. Lo colores pueden ser rojo, azul y verde que se representan con los números 1, 2 y 3 respectivamente. En la Tabla 32 se muestra un ejemplo de los datos iniciales de los objetivos.

Tabla 32

Ejemplo de los datos iniciales de los objetivos.

X	Y	Color
90	80	1[a]
45	150	2
15	60	3
0	200	1

[a] 1: Rojo, 2: Azul, 3: Verde

Una vez que el planificador ha recibido la orden de iniciar, crea 3 vectores *R*, *A* y *V* que contienen el número de fila correspondiente a los objetivos con los colores rojo, azul y verde respectivamente. Esta distinción se realiza mediante la lógica mostrada en la Figura 52. La variable *itni* es el inicio del ciclo automático y la variable *itmd* es verdadero cuando está activado el modo automático. El primer ciclo *mientras* se ejecuta cada 200 milisegundos y recibe los datos de los objetivos y la prioridad de organización. El ciclo se ejecuta en espera de que se dé la señal de iniciar y el modo manual este desactivado. Una vez que termina el primer ciclo mientras, se recorre el vector de colores *Co* comparándolo con los posibles valores, se agrega a los vectores *R*, *A*, *V* el índice *i* que representa la ubicación del objetivo en la matriz.

```
inicio
mientras (itni == falso y itmd == falso) hacer:
  Leer nobjetivos
  Leer [Xo[],Yo[],Co[]]
  Leer P[Pr,Pa,Pv]
  esperar 200 milisegundos
fin mientras
Para i = 0 hasta nobjetivos hacer:
  Si (Co[i] == 1) hacer:
    R[fin] = i
  Si (Co[i] == 2) hacer:
    A[fin] = i
  Si (Co[i] == 3) hacer:
    V[fin] = i
  i++
fin para
fin
```

Figura 52 Distinción del color de los objetivos

Una vez construidos los vectores *R*, *A* y *V* se procede a formar un solo vector *O* que contenga los índices en el orden que deben ser organizados. Para ello se sigue la lógica mostrada en la Figura 53. El ciclo *Para* recorre el vector prioridad y evalúa su contenido. Se crean los vectores auxiliares *Pr*, *Sg* y *Tr* que contienen los índices de los objetivos que deben ser organizados primeros, segundos y terceros respectivamente. Una vez finalizado el ciclo se crea el vector *O*.

```
incio
Para i = 0 hasta 3 hacer:
  Si (P[i] == 1) hacer:
    Si (i == 0) hacer:
      Pr = R
    Si (i == 1) hacer:
      Sg = R
    Si (i == 2) hacer:
      Tr = R
  Si (P[i] == 2) hacer:
    Si (i == 0) hacer:
      Pr = A
    Si (i == 1) hacer:
      Sg = A
    Si (i == 2) hacer:
      Tr = A
  Si (P[i] == 3) hacer:
    Si (i == 0) hacer:
      Pr = V
    Si (i == 1) hacer:
      Sg = V
    Si (i == 2) hacer:
      Tr = V
fin para
O = [Pr Sg Tr]
fin
```

Figura 53 Ordenar los objetivos en función de su prioridad

Posteriormente se procede a asignar los objetivos a los robots siguiendo la lógica mostrada en la Figura 54. Se crea la variable *ind* que recorrerá el vector *O* asignando los objetivos a los robots. El ciclo *mientras* se ejecuta hasta que *ind* sea mayor o igual que el número de objetivos *nobjetivos*, lo que garantiza el cumplimiento de la tarea. Si algún robot está disponible, estado que se almacena en la variable *drbx* y *ind* es menor que *nobjetivos* se procede a asignar los valores de los vectores *Xo*, *Yo* y *Co* a las variables *Xrx*, *Yrx* y *Crx* que se enviaran a los robots, se asigna verdadero a la variable *arbx* que significa que se ha hecho una nueva asignación al robot x. Se espera 300 milisegundos y se entra en el ciclo *mientras* que se ejecuta hasta que la variable *edrbx* sea verdadero, lo que ocurre cuando el robot x deja el espacio de trabajo disponible para los demás robots. Si el robot x está encendido, lo que significa que cumplió con éxito su tarea, se aumenta en la variable *ind*, de lo contrario se asigna el mismo objetivo al siguiente robot disponible.

```
1  inicio
2  ind = 0
3  mientras (ind < nobjetivos) hacer:
4    Si(drbx == verdadero y ind < nobjetivos) entonces:
5      Xrx = Xo[O[ind]]
6      Yrx = Yo[O[ind]]
7      Crx = Co[O[ind]]
8      arbx = verdadero
9      esperar 300 milisegundos
10     mientras (edrbx == falso) hacer:
11       esperar 200 milisegundos
12     fin mientras
13     Si (enrbx == verdadero) entonces:
14       ind++
15 fin mientras
16 fin
```

Figura 54 Asignación de objetivos a los robots

Una vez realizada la asignación al robot, se procede a enviarle los datos, para ello se sigue la lógica mostrada en la Figura 55. El ciclo *mientras* se ejecuta siempre que el *ind* sea menor que *nobjetivos*, si ha habido una nueva asignación a un robot y el espacio de trabajo está disponible, se pone en falso a la variable *arbx* que se ejecute nuevamente el ciclo y se publica el *Xrx*, *Yrx* y

Crx en los tópicos correspondientes. Finalmente se publica un 1 en *pl/rbx/lb*, se espera 3 segundos y se publica 0 en el mismo tópico. Con lo cual, el robot recibirá la orden de iniciar la secuencia automática.

```
1  inicio
2  mientras (ind < nobjetivos) hacer:
3    Si (arbx == verdadero y ed == verdadero) hacer:
4      arbx = falso
5      Publicar Xrx en pl/rbx/xd
6      Publicar Yrx en pl/rbx/yd
7      Publicar Crx en pl/rbx/cl
8      Publicar 1 en pl/rbx/lb
9      esperar 3000 milisegundos
10     Publicar 0 en pl/rbx/lb
11 fin mientras
12 fin
```

Figura 55 Envío de la información de objetivos a los robots

El planificador debe determinar si los robots están disponibles y su el espacio de trabajo está siendo ocupado. Para ello sigue la lógica mostrada en la Figura 56. El ciclo *mientras* se ejecuta siempre. Cualquier robot x está disponible cuando el robot informe de su disponibilidad en la variable *drrbx* y se detecte como encendido. El espacio está disponible cuando el robot informe que ha dejado el espacio de trabajo mediante la variable *errbx* o no se detecte como encendido.

```
1  inicio
2  mientras (verdadero) hacer:
3    drbx = (drrbx y erbx)
4    edrbx = no(errbx y erbx)
5  fin mientras
```

Figura 56 Determinación de la disponibilidad del espacio de trabajo

El robot realiza la secuencia automática cuando se ha recibido la orden de liberar *lb* como se muestra en la Figura 57. El ciclo *mientras* se ejecuta cada 200 milisegundos hasta que *lb* sea verdadero. Una vez terminado el ciclo, se realizan las siguientes acciones:

- Ir a los puntos asignados por el planificador.
- Levantar la pieza.
- Ir a la esquina superior izquierda de la zona de tránsito
- Ir la zona de desecho en función del color de la pieza asignada.
- Soltar la pieza.
- Publicar 0 en *rbx/es* lo que informa al planificador que ha dejado el espacio de trabajo.
- Ir al punto en *y* correspondiente a la zona de tránsito.
- Ir a la coordenada inicial en *y*, *yi*.
- Ir a la coordenada inicial en *x*, *xi*.

```
1  inicio
2  mientras (lb == falso) hacer:
3    esperar 200 milisegundos
4  fin mientras
5    ir(xd,yd)
6    levantar()
7    ir(145,250)
8    Si(cl == 1)
9      ir (90,220)
10   Si(cl == 2)
11     ir (50,220)
12   Si(cl == 3)
13     ir (10,220)
14   soltar()
15   Publicar 0 en rbx/es
16   ir(xc,250)
17   ir(-50,250)
18   ir(-50,yi)
19   ir(xi,yi)
20 fin
```

Figura 57 Secuencia automática realizada por el robot

El robot determina su disponibilidad mediante la lógica mostrada en la Figura 58. Se calcula la distancia *d* del centro del robot hasta su punto inicial, si la distancia es menor a 3 centímetros y el robot no está moviéndose, es decir *mov* es falso, el robot está disponible.

```
1  inicio
2  dx = |xc-xi|
3  dy = |yc-yi|
4  d = raiz(dx^2+dy^2)
5  Si (d<3 y mov == falso) entonces:
6    dis = verdadero
7  Si no:
8    dis = falso
9  fin
```

Figura 58 Disponibilidad del robot

CAPÍTULO 4

PRUEBAS Y RESULTADOS

El capítulo presenta el protocolo de pruebas que se siguió para determinar la funcionalidad del proyecto. El cual consistió en tres etapas, la evaluación de los robots, la evaluación de la arquitectura y la evaluación del control colaborativo.

4.1. Primera Etapa: Evaluación de los Robots

Las pruebas de los robots consistieron en evaluar la navegación de los mismos. Un valor adecuado en el error del posicionamiento de los robots influye en el desempeño del sistema en general. El método usado para esta evaluación consiste en tres pruebas para cada robot, con diez ensayos cada una. Las cuales consistieron en enviar a los robots a las coordenadas *(200,0)*, *(0,200)*, *(200,200)* en centímetros. Los resultados se presentan en la tabla IV.

Tabla 33

Resultado en el posicionamiento de los robots en los ejes X,Y

Robot	**(200,0) [cm]**	**(0,200) [cm]**	**(200,200) [cm]**
1	(199.12,0)	(0,199.60)	(195.55,198.65)
2	(199.42,0)	(0,199.05)	(197.15,197.05)
3	(199.27,0)	(0,199.36)	(196.35,197.85)
Promedio	(199.27,0)	(0,199.325)	(196.35,197.85)
Error [%]	0.365	0.337	1.449

La navegación de los robots tiene un margen de error menor a 2% para un recorrido de 200 centímetros, lo que asegura la funcionalidad del sistema. Los errores se acumularán para distancias mayores debido a las características inherentes de la estimación por odometría.

4.2. Segunda Etapa: Arquitectura IoT

4.2.1. Evaluación de criterios IoT

A. Tecnología distribuida: El protocolo MQTT se basa en TCP/IP, tecnología disponible para cualquier dispositivo con conexión a internet. La presente arquitectura une tecnología de National Instrument, Linux y Windows.

B. Interacción de objetos conectados: La estructura cliente – servidor y el intercambio de mensajes mediante publicación – subscripción de MQTT garantiza la interacción de los objetos conectados. Para el movimiento de un miembro del sistema Multi-Robot, es necesaria la interacción de Interfaz – Planificador – Robot.

C. Seguridad: El protocolo MQTT permite restringir la conexión bajo dos verificaciones: Usuario y contraseña. A demás de disponer del puerto seguro 8883 para su comunicación (Andy, Rahardjo, & Hanindhito, 2017). Ambas funciones se implementaron en la arquitectura desarrollada.

D. Escalabilidad: El bróker MQTT no depende del número de dispositivos conectados, su limitante se liga a la transferencia y procesamiento permitidos por el servicio de web hosting. Con la inclusión del planificador y su creación dinámica de tópicos, el sistema es fácilmente escalable con enfoque en los sistemas Multi-Robot.

E. Eficiencia Energética: MQTT es una versión de http ligera que ha permitido reducir significativamente el costo energético, sin embargo, sistemas como RFID superan a esta arquitectura en este aspecto. El desempeño de la arquitectura en función de criterios IoT se resume en la Tabla 34.

Tabla 34

Desempeño de la arquitectura en función de criterios IoT

Criterio	Desempeño		
	Bajo	***Medio***	***Alto***
1			X
2			X
3			X
4			X
5		X	

4.2.2. Tiempo de Conexión a la Red

El tiempo de envío y recepción de los mensajes es el factor determinante en la velocidad del sistema, para evaluar el tipo de conexión a internet adecuado para el desempeño del sistema se publicó en el bróker MQTT y se suscribió al mismo tópico desde un computador conectado a internet mediante la red de la Universidad de las Fuerzas Armadas - ESPE cuya velocidad es de 30Mbps de subida y bajada, conectado a una red celular 3G con una velocidad de 1Mbps de subida y bajada, y a una conexión de un proveedor privado con una velocidad de 6 Mbps de subida y bajada. Se midió el tiempo entre el envío y recepción del mensaje en diez ensayos. Los resultados se muestran en la Tabla 35.

Tabla 35

Tiempo promedio de envío y recepción de mensajes desde diferentes redes

Red	**Tiempo promedio [ms]**
Universidad	128.8
Privada	121.6
Celular	256.2

El menor tiempo de respuesta fue en la red conectada a internet mediante un proveedor privado. Las seguridades y el número dinámico de conexiones en la red universitaria reducen su velocidad a pesar de ser teóricamente mayor. La red celular tuvo el peor desempeño debido a su velocidad y tecnología.

4.2.3. Tiempo de cambio de Estado

La información sobre los robots encendidos permite al sistema tener datos actualizados sobre las unidades disponibles. El tiempo de detección entre los estados Apagado y Encendido influyen en las decisiones que se pueden tomar para su uso.

Se evalúa la velocidad de respuesta en los cambios de estado. Para lo cual se midió el tiempo desde que el robot se conectó a la red hasta que se visualizó su conexión en el planificador. Los resultados se muestran en la Tabla 36. Los datos corresponden al promedio de diez mediciones mediante una conexión a internet de 6 Mbps de subida y bajada.

Tabla 36

Tiempo de respuesta de las señales de encendido y apagado de los robots

Numero de robots	Tiempo encendido [s]	Tiempo apagado [s]
1	4.27	7.16
2	5.22	7.21
3	5.24	7.58
Promedio	4.91	7.32

El tiempo promedio de cambio de estado entre encendido y apagado es menor a 10 segundos. Por lo cual, las decisiones que necesiten del número disponible de robots se limitan a este intervalo de tiempo.

4.2.4. Costo Computacional

El uso en de los recursos computacionales de la máquina virtual influyen en la eficiencia y costo del sistema. Para determinar el costo computacional requerido por tareas típicas del sistema se procedió a enviar al robot uno a la coordenada *(200,200)* desde la interfaz en un computador mientras se monitoreó su posición desde un dispositivo móvil. Se registró el uso del CPU, Disco y ancho de banda público de la máquina virtual. El CPU de la máquina virtual es Intel Xeon E5-2650L v3 con velocidad máxima de 1.80 GHZ. Los resultados se muestran en la Tabla 37. El uso

de los recursos computacionales en la máquina virtual fueron mínimos. Por lo cual, la inclusión de nuevos miembros a la red es posible.

Tabla 37
Uso de los recursos de la máquina virtual

CPU [%]	Disco [B/s]	Ancho de banda público [Kb]
1.3	3.8	31.4

4.3. Tercera Etapa: Funcionalidad Colaborativa

Para la tercera etapa se realizaron pruebas al control colaborativo. Al realizar la tarea colaborativa se evalúo los casos de objetivo cumplido y no cumplido, desconexión y conexión de un robot y el tiempo de ejecución de la tarea.

4.3.1. Objetivo Cumplido

La prueba de objetivo cumplido consistió en disponer de tres objetivos con tres robots y permitir un funcionamiento sin fallos. La prueba se repitió diez veces alterando la prioridad y color de los objetivos para cada ensayo, manteniendo su ubicación constante. El comportamiento esperado es que los robots organicen las piezas en función de su color y prioridad, a cada robot se le debe asignar el siguiente objetivo a cumplir. Una vez terminada la tarea, los robots deben terminar inmóviles en su zona inicial. En todas las pruebas el comportamiento fue el esperado, por lo cual el resultado fue el 100% de cumplimiento.

4.3.2. Objetivo no cumplido

La prueba de objetivo no cumplido consistió en disponer de tres objetivos con tres robots y, una vez asignada la pieza, evitar que se cumpla al menos una vez. La prueba se repitió diez veces alterando la prioridad y color de los objetivos para cada ensayo, manteniendo su ubicación constante. El comportamiento esperado es que los robots organicen las piezas en función de su

color y prioridad, a cada robot se le debe asignar el objetivo a cumplir, incluyendo aquellos que se asignaron y no se cumplieron. Una vez terminada la tarea, los robots deben terminar inmóviles en su zona inicial. En todas las pruebas el comportamiento fue el esperado, por lo cual el resultado fue el 100% de cumplimiento.

4.3.3. Desconexión de robot

La prueba de desconexión de robot consistió en disponer de tres objetivos con tres robots y, una vez iniciada la tarea, desconectar a un robot permanentemente. La prueba se repitió diez veces alterando el robot desconectado, la prioridad y color de los objetivos para cada ensayo, manteniendo su ubicación constante. El comportamiento esperado es que los robots que permanecen encendidos organicen las piezas en función de su color y prioridad, a cada robot se le debe asignar el objetivo a cumplir. Una vez terminada la tarea, los robots deben terminar inmóviles en su zona inicial. En todas las pruebas el comportamiento fue el esperado, por lo cual el resultado fue el 100% de cumplimiento.

4.3.4. Conexión de robot

La prueba de conexión de robot consistió en disponer de tres objetivos con dos robots y, una vez iniciada la tarea, conectar al robot restante. La prueba se repitió diez veces alterando el robot a conectar, la prioridad y color de los objetivos para cada ensayo, manteniendo su ubicación constante. El comportamiento esperado es que el robot que ha sido encendido, en conjunto con los otros dos, organicen las piezas en función de su color y prioridad. Una vez terminada la tarea, los robots deben terminar inmóviles en su zona inicial. En ocho de las diez pruebas el comportamiento fue el esperado, por lo cual el resultado fue el 80% de cumplimiento.

El resultado de esta prueba se debe a la iteración en la que se encuentre el algoritmo de control para la asignación de objetivos, el tiempo de conexión y detección del robot encendido. Por lo

cual, si el planificador se ha saltado la asignación del robot no encendido, en función de la iteración de control, puede asignar los tres objetivos a los dos robots encendidos inicialmente. Sin embargo, la tarea se cumplió siempre.

4.3.5. Tiempo de cumplimiento

La prueba de tiempo de cumplimiento consistió en tres ensayos, en cada ensayo se dispuso de tres objetivos. Se midió el tiempo de cumplimiento de la tarea cuando se ejecutó con uno, dos y tres robots. No se introdujeron fallos intencionales al sistema. Los resultados se muestran en la Tabla 38.

Tabla 38
Resultados del tiempo de cumplimiento

Robots	Tiempo [min]
1	4.7
2	3.2
3	2.7

CAPÍTULO 5

CONCLUSIONES, RECOMENDACIONES Y TRABAJOS FUTUROS

5.1. Conclusiones

- Se diseñó una arquitectura IoT basada en MQTT para robótica colaborativa. La cual está compuesta por un subsistema Multi-Robot y un subsistema arquitectura IoT integrados mediante un planificador automático. Los robots son capaces de ubicarse en una zona de trabajo de seis metros cuadrados, manipular objetos demostrativos y evitar choques frontales. La arquitectura IoT hace uso del protocolo MQTT con un bróker accesible desde internet al igual que la interfaz de usuario. El planificador automático es el encargado de canalizar la información de la interfaz hacia los robots y realizar el control colaborativo.
- La evaluación de los criterios IoT para la arquitectura implementada reflejaron un desempeño alto en tecnología distribuida, interacción entre objetos, escalabilidad, seguridad y un desempeño medio en eficiencia energética. Lo cual implica que el proyecto desarrollado está en los marcos adecuados para IoT.
- Se determinó que el enlace de la arquitectura IoT mejora si la conexión a internet es por medio de una red privada. El tiempo de conexión mediante esta red es un 5.6% menor con respecto a la red de la universidad y 98.9% con respecto a una red celular.
- Se desarrolló un planificador automático clásico que permitió la canalización de información del usuario hacia los robots, la detección de las unidades disponibles en el sistema y el control colaborativo centralizado. Los estados para el control colaborativo se basaron en la aplicación. La cual consistió en trasladar piezas de madera en función de su color y la prioridad determinada por el usuario.

- El desempeño en la ejecución de la tarea por parte del sistema Multi-Robot refleja un control colaborativo adecuado que garantiza la clasificación de piezas en los casos de objetivo cumplido, objetivo no cumplido, conexión de un robot y desconexión de un robot.
- El costo computacional mínimo por parte del bróker MQTT que requiere la ejecución de la aplicación, extiende la operabilidad del sistema a problemas más complejos.

5.2. Recomendaciones

- El uso de un sistema híbrido de odometría y navegación inercial para el posicionamiento de robots móviles se recomienda cuando la estimación en la posición no sea crítica, especialmente para desarrollos de bajo costo y sobre plataformas con escasa capacidad de procesamiento.
- La elección del tipo de servicio para desarrollos IoT debe evaluarse en función del destino final de la aplicación y el número de dispositivos que han de conectarse a la nube. En lineamientos generales, se recomienda el uso de plataforma como servicio para aplicaciones introductorias que permitan beneficiarse de las pruebas gratuitas o en aplicaciones a gran escala en donde la magnitud justifique el costo elevado. Para proyectos medianos se recomienda hacer uso del modelo infraestructura como servicio. El cual reduce costos y es fácilmente escalable.
- La evolución de MQTT se ha basado en plataformas open source como RaspBerry, por ello se recomienda basar el desarrollo en este tipo de sistemas debido a que la documentación y el soporte por parte de la comunidad tecnológica es extenso.
- Se recomienda seguir una estructura ordenada para los tópicos en la comunicación mediante MQTT. Con lo cual se logra que la documentación y el seguimiento de fallas se facilite.

5.3. Trabajos Futuros

La arquitectura IoT implementada junto con su aplicación para robótica colaborativa pueden servir para desarrollos futuros, entre los que se propone:

- Incluir a objetos dentro de la red, de modo que elementos característicos del entorno se identifiquen y comuniquen con el sistema Multi-Robot para brindarle información sobre su ubicación, alarmas y variables de interés (temperatura, humedad, iluminación, etc.).
- Mejorar la manipulación del actuador haciendo uso de balizas de acercamiento al objetivo o mediante visión artificial.
- Mejorar la corrección en la estimación de la posición y orientación mediante visión artificial.
- Mejorar el planificador automático para permitir el desempeño del sistema en zonas no determinísticas y con la inclusión de obstáculos fijos.
- Combinar diferentes redes de IoT, especialmente incluir RFID, la cual es una tecnología de bajo costo energético y alta escalabilidad.

ANEXO 1

MANUAL DE USUARIO

El presente manual detalla el proceso de adquisición, instalación y puesta en marcha de la arquitectura IoT. Adicionalmente, se describe el procedimiento para agregar otro dispositivo a la red.

A. Proceso de adquisición e instalación

El primer paso consistió en la adquisición del servicio de Digital Ocean en el cual se creó una máquina virtual que fue usada como bróker. Para este proceso se precedió a crear una cuenta de usuario cuyo proceso se detalla en la Figura 59.

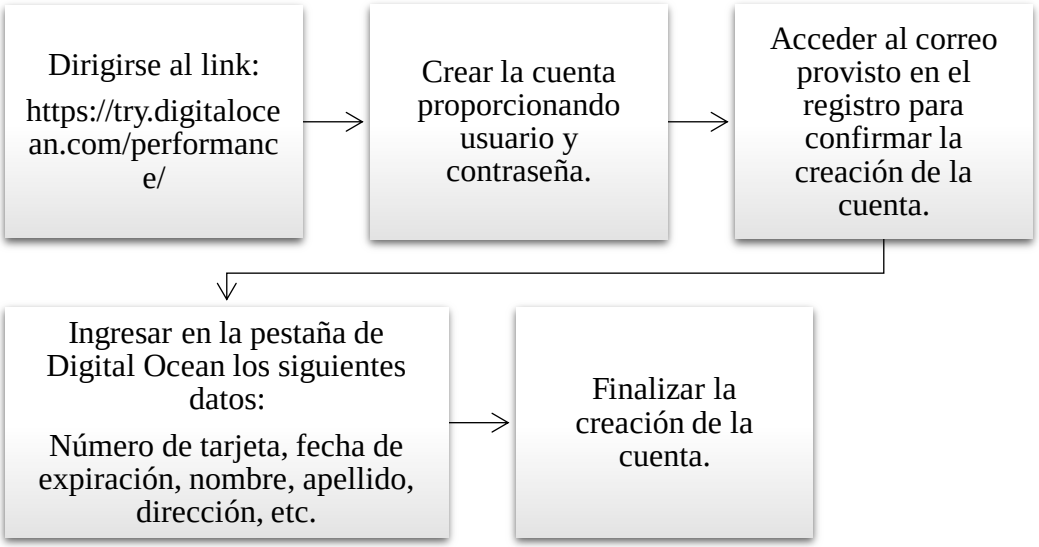
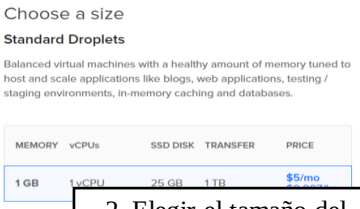


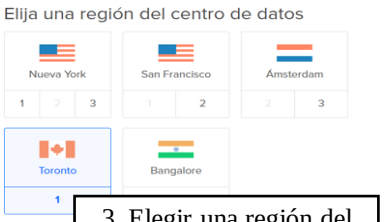
Figura 59 Proceso de creación de una cuenta en Digital Ocean

Una vez adquirida la cuenta en Digital Ocean, el siguiente paso consistió en la creación de un droplet, nombre que Digital Ocean da a las máquinas virtuales que se ejecutan en sus servidores. Para la creación del droplet se requiere de una llave, la cual puede ser creada en cualquier software, en este caso se usó el servicio de *keygen* proporcionado por MobaXterm (Hung, Kristiyanto, Lee, & Yeung). El proceso de creación del droplet se muestra en la Figura 60 .

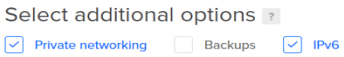
1. Elegir el S.O. del servidor.



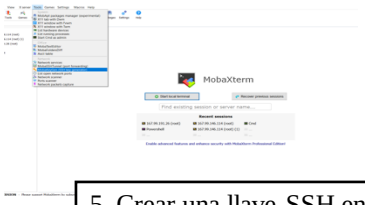
2. Elegir el tamaño del droplet.



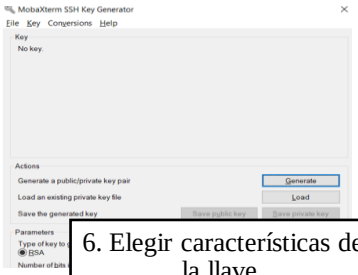
3. Elegir una región del centro de datos.



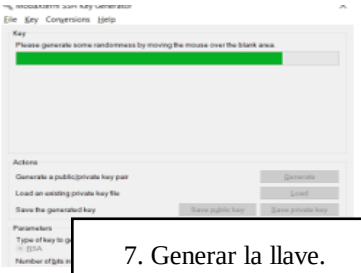
4. Seleccionar opciones adicionales.



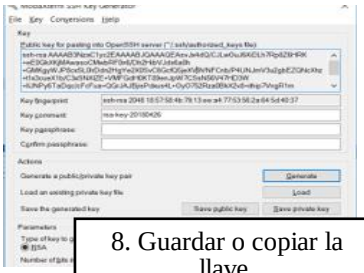
5. Crear una llave SSH en MobaXterm.



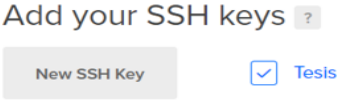
6. Elegir características de la llave.



7. Generar la llave.

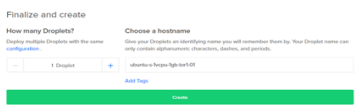


8. Guardar o copiar la llave.



9. Agregar la llave en Digital Ocean.

10. Pegar la llave creada.



11. Finalizar y crear el droplet.

12. Droplet creado con éxito.

Figura 60 Proceso de creación de la máquina virtual "*Droplet*"

El proveedor asignó la dirección IP pública al droplet, con la cual se puede acceder a la máquina virtual desde el internet como se muestra en la Figura 61.

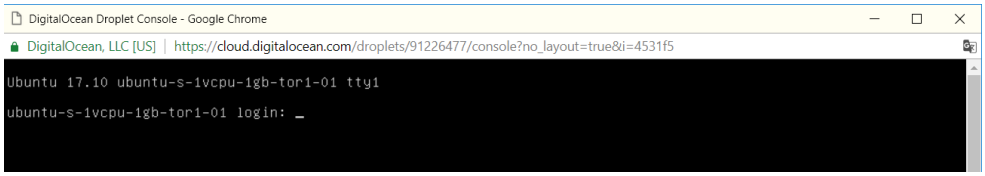


Figura 61 Consola de la máquina virtual

Una vez que se ha creado la máquina virtual se procede a encenderla e instalar el bróker MQTT siguiendo los pasos que se muestran en la Figura 62 mientras que para el desarrollo de la interfaz de usuario se instaló la herramienta Node-RED como se muestra en la Figura 63.

- ***MQTT***

Actualizar paquetes

sudo apt-get update
sudo apt-get upgrade

Instalar Mosquito

sudo apt-get install mosquitto
sudo apt-get install mosquitto-clients

Autentificación

touch passwd
sudo mosquitto_passwd /etc/mosquitto/passwd *usuario contraseña*

Reiniciar el servicio Mosquitto

service mosquitto restart

Configuración de mosquito

Crear el archivo monquitto.conf en el directorio mosquitto

nano /etc/mosquitto/mosquitto.conf

Ingresar la siguiente información

allow_anonymous false
password_file /etc/mosquittto/passwd

```
# Place your local configuration in /etc/mosquitto/conf.d/
#
# A full description of the configuration file is at
# /usr/share/doc/mosquitto/examples/mosquitto.conf.example

password_file /etc/mosquitto/passwd
allow_anonymous false
```

Cerrar el archive y reiniciar el servicio

service mosquitto restart

Figura 62 Proceso de Instalación y Autentificación de MQTT

- ***Node-RED***

Instalar Node.js

sudo apt-get install nodejs

Instalar gestor de repositorios de node.js

sudo apt-get install mosquitto

sudo apt-get install mosquitto-clients

Instalar NodeRED

sudo npm install -g --unsafe-perm node-red

Correr NodeRED en segundo plano

screen node-red

Figura 63 Proceso de Instalación de Node-RED

B. Puesta del proyecto

Para poner en funcionamiento al kit de robots que forman parte de la arquitectura IoT se deben seguir los siguientes pasos:

- Iniciar el planificador automático mediante el ejecutable *planificador.exe* como se muestra en la Figura 64. En la parte inferior izquierda se debe ingresar los datos de los objetivos (número, ubicación y color).

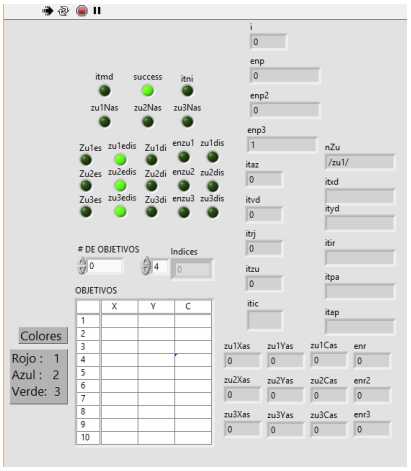


Figura 64 Planificador

- Energizar las tarjetas NI MyRIO mediante las baterías montadas sobre los robots. Cada tarjeta cuenta con 6 leds clasificados en dos grupos: El primer grupo contiene 2 indicadores que representan el encendido del robot y su adecuada conexión a la red WiFi, el segundo grupo de leds está compuesto de 3 indicadores que fueron usados para representar la publicación de mensajes al bróker, la suscripción a los tópicos correspondientes en el servidor y el funcionamiento adecuado del giroscopio, respectivamente.
- Abrir NodeRED desde cualquier dispositivo con acceso a internet, la dirección de la interfaz se compone de la IP del bróker, dos puntos (:), el puerto por el cual se ejecuta NodeRED, barra oblicúa (/) y *ui*. En el caso del proyecto realizado es: 159.89.114.195:8080/ui.
- En la ventana de control existen dos modos de funcionamiento, manual y automático. En el modo manual se puede manipular la posición del robot, la inclinación y apertura del actuador. En modo automático existen controles para establecer la prioridad de recolección de objetivos y un botón para iniciar la tarea colaborativa.
- En la ventana de monitoreo se puede visualizar el estado de los robots y su posición en tiempo real.

C. Paso a paso para la inclusión de un dispositivo a la arquitectura IoT

En este ejemplo se incluirá a un sensor de temperatura en la arquitectura IoT. Los materiales necesarios son:

- 1 tarjeta de desarrollo NodeMCU ESP8266
- 1 sensor de temperatura
- Jumpers

Los pasos para incluir al sensor de temperatura en la arquitectura IoT son los siguientes:

- Preparar el entorno IDE de Arduino para la programación de la tarjeta. El primer paso consiste en ingresar a la sección preferencias como se ilustra en la Figura 65.

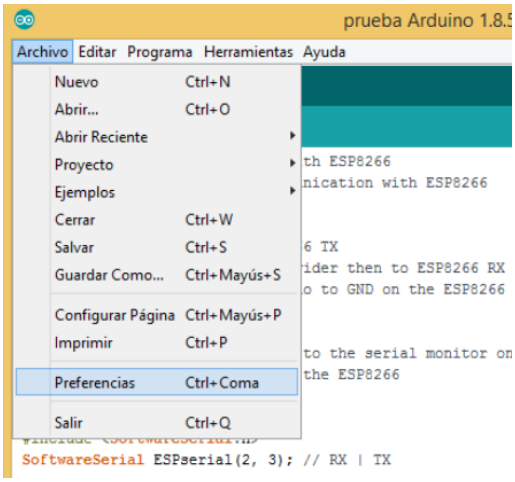


Figura 65 Sección preferencias IDE de Arduino

- Pegar el link "*http://arduino.esp8266.com/stable/package_esp8266com_index.json*" en gestor de URLs como se muestra en la Figura 66.

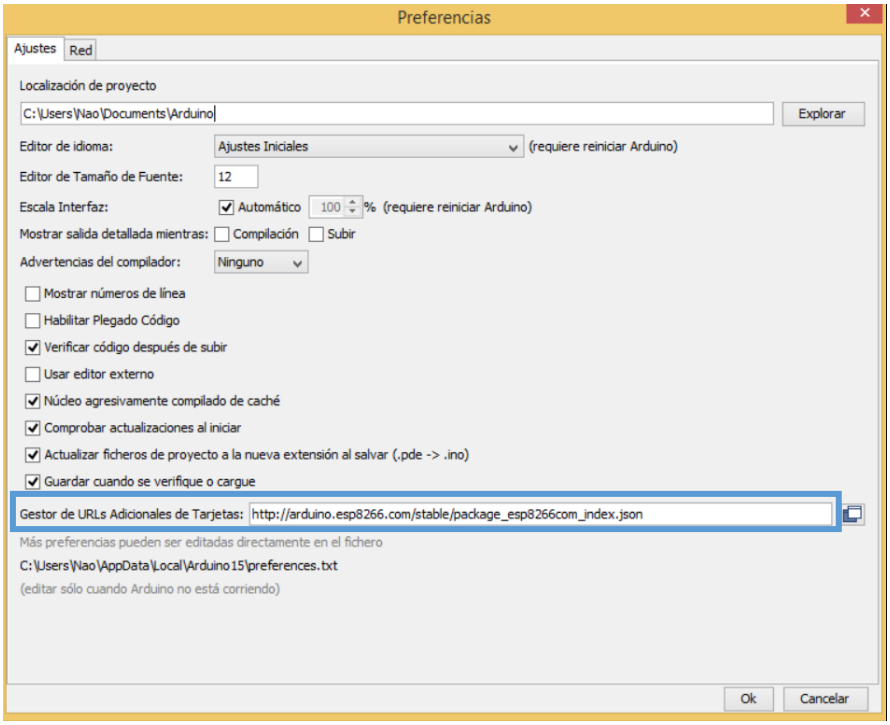


Figura 66 Ingreso del link en el gestor de URLs

- Ingresar al gestor de tarjetas para configurar la NodeMCU

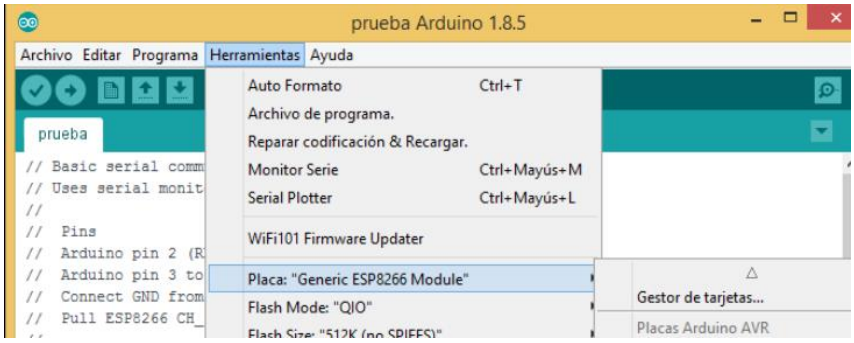


Figura 67 Gestor de Tarjetas del IDE de Arduino

- Seleccionar ESP8266 e instalar la versión más actual como se muestra en la Figura 68.

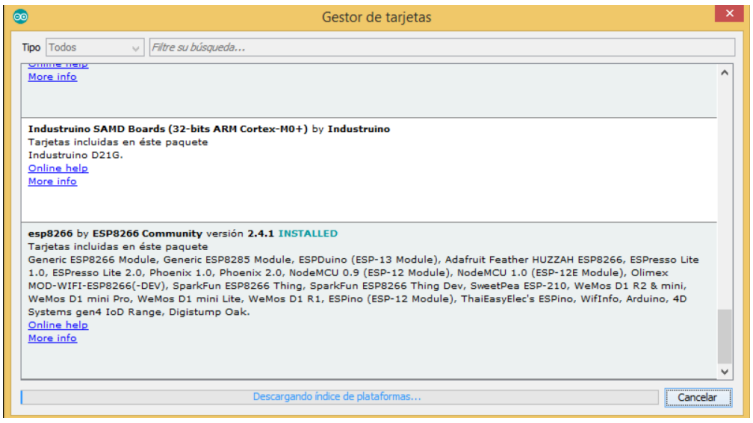


Figura 68 Selección de ESP8266 en el gestor de tarjetas

- Conectar el sensor a la tarjeta como se ilustra en la Figura 69.

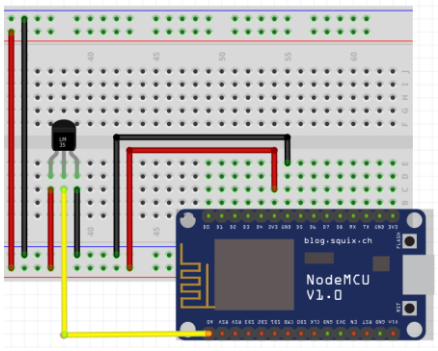


Figura 69 Conexión de la tarjeta NodeMCU

- Programar la tarjeta con el algoritmo mostrado en la Figura 70.

```
//Incluir las librerías necesarias para el uso de la tarjeta y el protocolo MQTT:
#include <ESP8266WiFi.h>
#include <Adafruit_MQTT.h>
#include <Adafruit_MQTT_Client.h>
//Definición de variables a usar
unsigned long previousMillis = 0;
int estado=0;
//Función MQTT_connect()
void MQTT_connect();
WiFiClient client;
Adafruit_MQTT_Client mqtt(&client,"159.89.114.195", 8883, “zu1”,”1111” );
Adafruit_MQTT_Publish temperatura=Adafruit_MQTT_Publish(&mqtt, "d1/temperatura");
//Inicialización
void setup() {
    Serial.begin(115200);

    WiFi.persistent(false);
    WiFi.mode(WIFI_STA);

    //Usuario y contraseña del servicio de internet al cual se conecta
    char __ssssid[] = "user";
    char __spasswd[] = "pass";
    WiFi.begin(__ssssid,__spasswd);

    //Conectar WiFi
    int retries = 0;
    while ((WiFi.status() != WL_CONNECTED) && retries<=70) {
        retries++;
        WiFi.begin(__ssssid,__spasswd);
    }

    //WiFi Conectado
    if (WiFi.status() == WL_CONNECTED) {
        estado=1;
    }

}

void loop() {

    //Conexión a MQTT
    MQTT_connect();
    mqtt.processPackets(10000);
```

CONTINÚA →

```
    //Toma de datos del sensor de temperatura
    unsigned long currentMillis = millis();
    if (currentMillis - previousMillis >= 500) {
        previousMillis = currentMillis;
        int temp=analogRead(A0);
        float milivoltios = (temp / 1023.0) * 3300;
        float celsius = milivoltios / 10;
        temperatura.publish(celsius);
    }

    //Desconexión de MQTT
    if(! mqtt.ping()){
        mqtt.disconnect();
    }
}
//Función para conectar a MQTT
void MQTT_connect() {
    int8_t ret;
    // Parar si ya se realizó la conexión.
    if (mqtt.connected()) {
        return;
    }
    //Intentos de conexión
    uint8_t retries = 10;
    while ((ret = mqtt.connect()) != 0) {
        mqtt.disconnect();
        delay(500);  // wait 5 seconds
        retries--;
        //Resetear tarjeta
        if (retries == 0) {
            while (1);
        }
    }
}
```

Figura 70 Programación en el IDE de Arduino

- El código se encarga de conectarse al bróker automáticamente y publicar la temperatura ambiente cada 500 milisegundos en el tópico *d1/temperatura*.

- Para la visualización se debe ingresar a NodeRED y crear una interfaz de usuario, para ello se usará un bloque de entrada de MQTT y un indicador tipo Gauge para la presentación. En la Figura 71 se muestra el bloque MQTT y su configuración. En la Figura 72 se muestra la conexión del bloque MQTT con el indicador y su presentación en la pantalla.

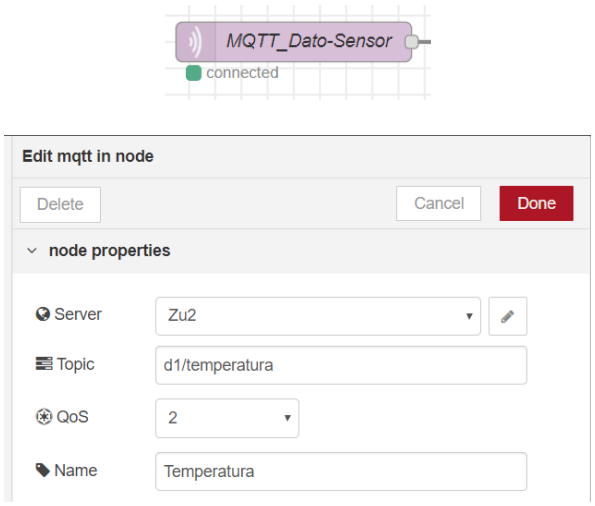


Figura 71 Configuración del bloque MQTT

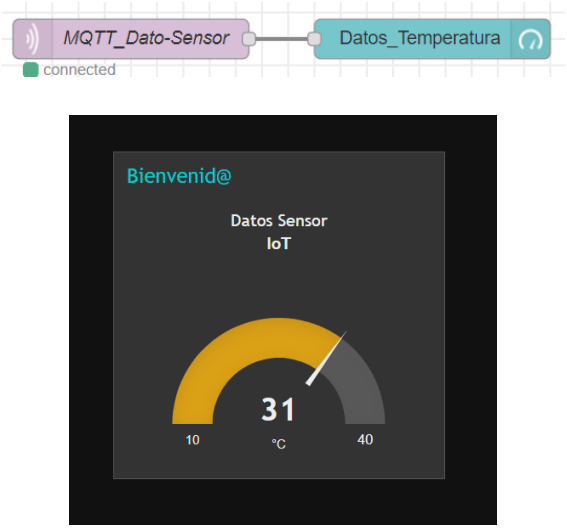


Figura 72 Conexión del bloque MQTT y resultado en la interfaz

BIBLIOGRAFÍA

Abasolo Aranda, S. E., & Carrera Paz y Miño, M. A. (2014). *Evaluación del modelo de referencia de "Internet of Things" (IoT), mediante la implantación de arquitecturas basadas en plataformas comerciales, open hardware y conectividad IPv6.* Sangolquí.

Álvarez, C. R., Aracil, R., & García, C. (2009). *Concepción, Desarrollo y Avances en el Control de Navegación de Robots Submarinos Paralelos: El Robot Remo-I.* MAracaibo: Revista Iberoamericana de Automática e Informática Industrial RIAI.

Andy, S., Rahardjo, B., & Hanindhito, B. (2017). *Attack scenarios and security analysis of MQTT communication protocol in IoT system.* Yogyakarta: IEEE.

Arévalo, M., & Pino, M. (2005). *Control por voz de un robot explorador tipo oruga.* Quito: Escuela Politécnica Nacional.

Ashton, K. (22 de Junio de 2009). That 'Internet of Things' Thing. *RFID Journal*, pág. 1.

Blackstock, M., & Lea, R. (2015). *Toward a Distributed Data Flow Platform for the Web of Things.* Seoul: IEEE.

Campoverde, P. E. (2017). *Desarrollo de dos robots para realizar trabajo cooperativo.* Sangolquí.

Castells, M. (2000). *Internet y la Sociedad Red.* Catalunya.

Castillo Merchán, H. A. (2016). *Análisis de la gestión de seguridad y fallos en internet de las cosas, usando el estandar 6LowPan.* Quito: UDLA.

Castro Heredia, J. (2014). *Uso del protocolo CoAP para la implementación de una aplicación domótica con redes de sensores inalámbricas.* Cartagena: etsit.

CIC, I. C. (20 de Diciembre de 2013). Santander, modelo español de “Smart City”. Santander, España.

Cisco. (24 de Enere de 2017). Personas + Procesos + Datos + Dispositivos = IoE.

Collina, M., Corazza, G. E., & Vanelli-Coralli, A. (2012). *Introducing the QEST broker: Scaling the IoT by bridging MQTT and REST.* Sydney: IEEE.

Cowen. (7 de Enero de 2018). mqtt-LabVIEW.

Cruz, H. (2008). *Una Introducción a los Robots Móviles.* Bogotá: Universidad Distrital Francisco Jose de Caldas.

Digillent. (2016). *PmodGYRO™ Reference Manual.* Pulman.

Espada Recarey, L. (2014). *Smart City Implantaciónd e un sistema inteligente de aparcamiento en espacios libres en la ciudad de Vigo.* Vigo: Ariel.

Evans, D. (2011). *The Internet of Things How the Next Evolution of the Internet Is Changing Everything.* San José: Cisco IBSG.

Ghallab, M., & Nau, D. (2004). *Automated Planning: Theory and Practice.* Morgan Kauffman Publishers.

Glen, M., & Moreno, J. (23 de Mayo de 2012). ZigBee.

González Jiménez, J., & Ollero Baturone, A. (1996). "Estimación de la posición de un robot móvil" Informática y Automática. *Asociación Española de Informática y Automática.*, 3-18.

Guoqiang, H., Wee Peng, T., & Yonggang, W. (2012). *Cloud robotics: architecture, challenges and applications.* Singapore: IEEE Network.

Haque. (10 de Noviembre de 2012). Pachube.

Hung, L.-H., Kristiyanto, D., Lee, S. B., & Yeung, K. Y. (s.f.). *GUIdock.* Washington: University of Washington.

Jones, M. (2015). *31150-MP MG995 High Speed Servo Actuator.* West Palm Beach: MPJA.

Joskowicz, J. (2014). *Historia de las Telecomunicaciones.* Uruguay.

Kantel, E., Tovar, G., & Serrano, A. (2010). Diseño de un Entorno Colaborativo Móvil para Apoyo al Aprendizaje a través de Dispositivos Móviles de Tercera Generación. *IEEE-RITA*, 146-151.

Kehoe, B., Patil, S., Abbeel, P., & Goldberg, K. (2015). *A Survey of Research on Cloud Robotics and Automation.* California: IEEE Transactions on Automation Science and Engineering .

Köhler, M., Wörner, D., & Wortmann, F. (2014). *Platforms for the Internet of Things - An Analysis of Existing Solutions.* Zurich.

Kumar, V., Daniela, R., & Sukhatme, G. S. (2004). *"Networked Robotics,".* Berlin Heidelberg: Springer Handbook of Robotics.

Larrañaga Fuerte, J. (2016). *IoT con IBM y NI.* País Vasco.

Leonard, J., & Durrant-Whyte. (1991). Mobile Robot Localization by Tracking Geometric Beacons. *IEEE Transaction on Robotics and Automation*, 376-382.

Lozano Feliú, M. A. (2017). *Estudio y desarrollo de una aplicación mediante comunicación WiFi.* Madrid.

Madakam, S., Ramaswamy, R., & Tripathi, S. (2015). *Internet of Things (IoT): A Literature Review.* Mumbai: Journal of Computer and Communications.

Mahmoud, R., Yousuf, T., Aloul, F., & Zualkernan, I. (2016). *Internet of Things (IoT) Security: Current Status, Challenges and Prospective Measures.* London, UK: IEEE.

Marroquin, A., Gomez, A., & Paz, A. (2017). *Design and implementation of explorer mobile robot controlled remotely using IoT technology.* Pucón: IEEE.

Mohanarajah, G., Hunziker, D., D'Andrea, R., & Waibel, M. (2014). *Rapyuta: A Cloud Robotics Platform.* Zurich: IEEE Transactions on Automation Science and Engineering.

Montoya, J., Pérez, C., Garnica, E., Salamanca, D., & Simanca, J. (2012). *Diseño y construcción de un robot para limpieza acuática.* Bogotá: Universidad de La Salle.

National Instruments. (s.f.). Robot Builder's Guide. *PITSCO Tetriz Prime for MyRIO*, 130.

Nava Díaz, S., Chavarría Juarez, G., Hervás Lucas, R., & Bravo Rodriguez, J. (2009). *Adaptabilidad de las tecnologías RFID y NFC a un contexto educativo: Una experiencia en trabajo cooperativo* . Tampico: IEEE.

Orbea, D., & Moposita, J. (2017). *Diseño y construcción de un prototipo UAV multirotor de mini escala con estructura aerodinámica de ala fija.* Sangolquí: ESPE.

Oussama Khatib, B. S. (2008). *Handbook of robotics.* Springer.

Parker, L. (2012). Decision making as optimization in multi-robot teams. *Distributed Computing and Internet Technology*, 35-49.

Pavón, J. (2012). *Protocolos y arquitecturas de aplicaciones en internet.* Madrid: UCM.

Peña Merizalde, J. L., & Suquillo Chuquimarca, G. E. (2016). *Estudio del modelo de referencia del Internet de las cosas (IoT),c on la implementación de un protoripo domótico.* Quito: EPN.

Portilla, B., & Guzmán, J. (2013). *Computación en la nube como modelo distribuido para la interaciión de plataformas robóticas.* Medellín: Universidad Nacional de Colombia, Medellín.

Primo Guijarro, Á. (2012). *Protocolo HTTP*.

Puertas, D. (2016). *DronePi: Construcción de un dron basado en Raspberry Pi.* Valencia: Universidad Politécnica de Valencia.

Pulido Cañabate, E. (2016). *Big data: ¿Solución o problema?* Madrid: Universidad Autónoma de Madrid. Fundación General.

Qureshi, R., Mohammad AlManna, Y., & Pasha Deshmukh, A. (2013). *Big Data: Growing pressure on global storage by data created on Social Networking Sites.* Jazan: ISSN.

Rabelo, J., González, M., Pérez, H., & Delgado, M. (2015). *Sistema de Notificaciones para la Plataforma de Desarrollo de Integración Continua de la distribución cubana de GNU/Linux, Nova.* La Habana: UCI.

Ramírez, G. (2003). *Método de aprendizaje simple para navegación de minirobots móviles rodantes.* Medellín: Universidad Nacional de Colombia.

Rodriguez, C., & Geovanny, F. (2015). *El Internet de las cosas y las consideraciones de seguridad.* Quito: PUCE.

Ruiz Libreros, E. (2013). *Robots M´oviles Colaborativos para el Transporte Autónomo de Objetos V´ıa Visi´on.* Ciudad de México: Universidad Nacional Aut´onoma de México.

San Millán Rodríguez, A. (2012). *Diseño, construcción y control de una serpiente robótica.* La Mancha: Universidad de Castilla.

Sha Yang Ye. (2014). *IG220053X00085R Datasheet.* Taoyuan.

STMicroelectronics. (2010). *MEMS motion sensor: ultra-stable three-axis digital output gyroscope.*

Tabango, R. D. (2014). *Desarrollo de un sistema de robótica colectiva con procesamiento centralizado entre dos robots humanoides Bioloid Premium.* Sangolquí.

Ulrich, K., & Eppinger, S. (2013). *Diseño y desarrollo de productos.* México: McGrawHill.

VDI-2206. (2014). *Design methodology for mechatronic systems.* Düsseldorf.

Vera Cañal, R. M. (2015). *Sistemas de planificación en robótica.* Madrid: Universidad Carlos III de Madrid.

Zambrano, A., Pérez, I., Plau, C., & Esteve, M. (2015). *Sistema Distribuido de Detección de Sismos Usando una Red de Sensores Inalámbrica para Alerta Temprana.* Valencia: Elsevier.