DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

TRABAJO DE TITULACIÓN, PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS E INFORMÁTICA

TEMA: ARQUITECTURA DE CONTROL Y MONITOREO DE ROBOTS PARA BÚSQUEDA Y RESCATE URBANO POR MEDIO DE TECNOLOGÍAS SERVERLESS DE LA NUBE

AUTOR: MEJÍA SIZONENKO, ALEXANDER VICENTE

DIRECTOR: ING. MARCILLO PARRA, DIEGO MIGUEL

SANGOLQUÍ

2020

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

CERTIFICACIÓN

Certifico que el trabajo de titulación, **"Arquitectura de control y monitoreo de robots para búsqueda y rescate urbano por medio de tecnologías serverless de la nube"** fue realizado por el señor **Mejía Sizonenko, Alexander Vicente** 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, **13 de enero de 2020**

Firma:

Ing. Marcillo Parra, Diego Miguel

CC. 1710802925

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

AUTORÍA DE RESPONSABILIDAD

Yo, **Mejía Sizonenko, Alexander Vicente** , declaro que el contenido, ideas y criterios del trabajo de titulación: **Arquitectura de control y monitoreo de robots para búsqueda y rescate urbano por medio de tecnologías serverless de la nube** es de mi 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, **13 de enero de 2020**

Firma:

Mejía Sizonenko, Alexander Vicente

CC. 1717815367

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

AUTORIZACIÓN

Yo, **Mejía Sizonenko, Alexander Vicente** autorizo a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación: **Arquitectura de control y monitoreo de robots para búsqueda y rescate urbano por medio de tecnologías serverless de la nube** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de mi responsabilidad.

Sangolquí, **13 de enero de 2020**

Firma:

Mejía Sizonenko, Alexander Vicente

CC. 1717815367

DEDICATORIA

Para Dimitri y Nikita.

AGRADECIMIENTO

Agradezco a mis padres, familia, amigos, mis profesores, a la Universidad de las Fuerzas Armadas y al Club de Software ESPE. Sin la ayuda y el apoyo de ellos, no podría llegar a ser la persona que soy ahora, y este trabajo no podría ser posible.

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

Cada año ocurren diferentes acontecimientos que afectan grandes grupos poblacionales; ya sea por desastres naturales, desastres industriales o accidentes humanos. Las operaciones de búsqueda y rescate son un elemento fundamental para asegurar y proteger vidas humanas y bienes materiales frente a sucesos de este tipo. Las operaciones de búsqueda y rescate han presentado grandes avances, entre estos el uso de robots. Los robots usados aún son complejos, difíciles de comunicar y requieren de tecnología especializada. Este trabajo propone estructurar una arquitectura de control y monitoreo de robots de búsqueda y recate urbano por medio de tecnologías serverless basadas en la nube para disminuir la complejidad técnica y operativa bajo costo. Al poner a prueba esta arquitectura, se encontró que rinde adecuadamente durante el control y monitoreo, y presenta un despliegue simple y rápido; incluso con ancho de banda limitado, pero con una buena señal de conexión. Tiene latencias de entre 200ms con alta intensidad de señal. Aun así, se encontraron limitantes en conexiones wifi, realizar video streaming y determinación de costo preciso exactos de uso. Por lo tanto, es necesario expandir la investigación para mejorar las capacidades de la arquitectura y poder probarla en ambientes reales.

PALABRAS CLAVE

- **OPERACIONES DE BÚSQUEDA Y RESCATE URBANO**
- **SERVERLESS COMPUTING**
- **ROBOT**

ABSTRACT

Every year several events take place that have immense effect in large group of the world population. These events occur due to natural disasters, industrial disasters or human provoked accidents. Search and rescue operations are a fundamental element to ensure and protect human lives and material goods from this kind of events. Search and rescue operations have made great progress, most notably the use of robots. These robots are still very complicated, have communication challenges and require specialized technology to work properly. This work proposed to structure an architecture for the control and monitoring of urban search and rescue robots with the usage of cloud-based serverless technologies to reduce technical and operational complexity at a low cost. When testing this architecture, it was found that it has a proper performance during control and monitoring; it also presents a simple and rapid deployment; even with limited bandwidth, but with the necessity of a good connection signal. Usual latency was of 200ms with good quality signal. Although an adequate performance was found, limitations were found in the need of Wi-Fi connections, video streaming and accurate cost of use determination. Therefore, it is necessary to expand research improving the architecture's capabilities and testing it in real environments.

KEY WORDS

- **URBAN SEARCH AND RESCUE OPERATIONS**
- **SERVERLESS COMPUTING**
- **ROBOT**

CAPÍTULO I

INTRODUCCIÓN

1.1 ANTECEDENTES

Cada año ocurren una gran cantidad de sucesos que afectan a grandes grupos poblacionales por medio de desastres naturales, accidentes industriales a gran escala, accidentes humanos en grandes proporciones (Nurmi & Tarkoma, 2017). Tales acontecimientos requieren de una respuesta rápida que permita salvar la mayor cantidad de vidas y la reducción mínima de daños materiales.

Las operaciones de búsqueda y rescate son aquellas que se encargan de proveer auxilio a individuos, poblaciones, maquinaria, vehículos, etc. durante situaciones de alto riesgo que generan una situación de peligro (Goverment of Canada, 2018). Por lo tanto, situaciones de desastre, tales como las ya mencionadas en párrafos anteriores requieren de la presencia de operaciones de búsqueda y rescate que permitan ayudar a las personas afectadas, recuperar bienes y reducir en la mayor medida posible el alcance de los daños provocados. Para llegar a estas metas, se debe seguir diferentes procesos y metodologías para obtener los mejores resultados posibles. Según la INSARAG (International Search and Rescue Advisory Group), una organización de la ONU, existe un procesos a llevar a cabo en Búsqueda y Rescate Urbanos que sigue las siguientes fases: Preparación, Movilización, Operaciones, Desmovilización y Post-Misión (INSARAG, n.d.-a). Además de esto, se presentan guías de cómo manejar recursos humanos, vehículos y maquinaria para llevar a cabo exitosamente operaciones de búsqueda y rescate (INSARAG, n.d.-b).

Debido al gran esfuerzo que supone una operación de búsqueda y rescate urbano, a lo largo del tiempo se han desarrollado diversas tecnologías y técnicas que han ayudado a agilizar estas operaciones y tener resultados más exitosos. Estas herramientas tecnológicas, ayudan a los equipos de búsqueda y rescate con movilización, ubicación, reconocimiento, mapeo, atención de primeros auxilios, etc.(INSARAG, n.d.-b). Dentro de todas las herramientas tecnológicas que permiten a los equipos de búsqueda a ser efectivos son el uso de robots (Cubber et al., 2017). Estos robots ayudan a los operadores humanos a tener una mejor percepción de la situación y por lo tanto una acción más efectiva (Cubber et al., 2017). Aunque el uso de robots es una tendencia que no es del todo nueva, no ha tenido una aplicación a gran escala (Cubber et al., 2017). Entre los mayores retos que existen en la actualidad son la implementación de infraestructuras, costos de operación y especialización de los robots; que hacen que diversos robots de búsqueda y rescate sean difíciles de implementarlos y solo funcionen en situaciones especifican o solo puedan implementarse por organizaciones con amplia cantidad de recursos (Cubber et al., 2017). Proyectos como ICARUS, buscan impulsar el uso de la robótica en operaciones de búsqueda y rescate (ICARUS, n.d.).

Debido al claro potencial de la robótica en operaciones de búsqueda y rescate urbano, en los últimos años ha existido un constante flujo de investigaciones y tendencias hacia aplicaciones robóticas en búsqueda y rescate; los elementos notables son la implementación de biobots y el uso de infraestructuras en la nube. En el primer caso, se puede destacar como ejemplo la investigación de (Latif, Whitmire, Novak, & Bozkurt,

2016) se experimentó con la implantación de sensores de sonido en cucarachas y otros medios para el control de las funciones motoras de estas; así, se lograr tener una forma de mapear una zona de difícil acceso dentro de una operación de búsqueda y rescate. En el otro caso, la investigación de (Mouradian, Yangui, & Glitho, 2018) se propone un modelo que permite la entrega de robots como un servicio basado en la nube donde se puede generar comunicación, despliegue y customización de robots a base de las necesidades de una operación de búsqueda y rescate urbano.

1.2 PLANTEAMIENTO DEL PROBLEMA

El uso de robots para operaciones de búsqueda y rescate tienen un gran potencial y han mostrado un beneficio claro dentro de las operaciones donde se han usado. Usos notables de estos robots se han dado en eventos como el tsunami en Japón durante 2011 o los terremotos en Mid Niigata en 2004 (Pelka et al., 2014). La implementación de robots permite un mejor estudio del entorno y por ende una mejor respuesta ante la posible situación de desastre; aun así, existen diversas dificultades durante la operación de estos robots, como:

- Implementación y despliegue lento de robots de búsqueda y rescate (Cubber et al., 2017).
- Autonomía y capacidades limitadas debido al nivel de procesamiento actual de los robots (Cubber et al., 2017).
- Comunicación limitada entre robots (Cubber et al., 2017).
- Dificultad de integración con equipos de búsqueda y rescate (Cubber et al., 2017).

- Control complejo que requiere de un equipo especializado (Cubber et al., 2017).

Una de las causas más claras para estos problemas son las infraestructuras sobre lo que estos robots funcionan y se comunican. Métodos tradicionales de comunicación dificultan la comunicación y la integración con diferentes dispositivos. Generalmente esto se debe a que los robots carecen de una integración en red o carecen de una infraestructura que permitan procesar información. Esto genera el problema de que los sistemas de control y monitoreo para los robots de búsqueda y rescate urbano podrían resultar complicados de manejar, costosos y requieren de una amplia variedad de hardware, software e infraestructura. Estas dificultades son elementos que reducen la efectividad de estos robots y a su vez reducen su capacidad de uso dentro de operaciones de búsqueda y rescato urbano. Provocando que solo puedan ser usados en escenarios específicos y por gente especializada.

Diferentes esfuerzos se han realizado para acortar este problema, más notablemente el uso de la nube (Mouradian et al., 2018). Estas investigaciones proponen arquitecturas que permiten hacer uso de recursos de la nube para procesar información, comunicar y controlar robots. Resultan muy útiles y un tanto efectivas para dar más versatilidad a los robots; pero, resultan complejas de manejar, difícil de implementar y requieren de una amplia gama de recursos (Mouradian et al., 2018). También empiezan a surgir retos con la cantidad de datos transmitidos, las velocidades necesarias y accesos a redes de comunicación (Nakamura et al., 2018). Por último, existen retos que se forman con la dificultad de presentar medios de acceso a la información que recolecta estos robots requieren de estaciones bases y resulta difícil que equipos de rescate que se despliegan

en el campo de operación tengan un acceso completo a la información del robot de manera directa (Sharma, Young, & Eskicioglu, 2012).

1.3 OBJETIVOS

1.3.1 Objetivo General

Estructurar una arquitectura de control y monitoreo de robots de búsqueda y recate urbano por medio del uso de hardware, software y tecnologías serverless basadas en la nube para disminuir la complejidad técnica y operativa a un costo reducido en el manejo de estos robots.

1.3.2 Objetivos Específicos

i. Investigar herramientas, metodologías y modelos utilizados en las operaciones de búsqueda y rescate urbano para poder relacionarlas con las capacidades de las herramientas de la nube.

ii. Diseñar una arquitectura que permita el control y monitoreo de diversos robots de manera simultánea por medio de herramientas basadas en la nube y de tipo serverless.

iii. Implementar la arquitectura de control y monitoreo con un prototipo en un entorno controlado de tal manera que se pueda medir su funcionamiento y utilidad.

iv. Validar la arquitectura de control y monitoreo con pruebas de QoS, para así poder determinar sus limitaciones y su nivel de utilidad en una operación de búsqueda y rescate urbano.

1.4 JUSTIFICACIÓN

Sucesos tales como desastres naturales, accidentes industriales y accidentes humanos en gran escala, aunque relativamente extraños generan muchos daños y afectan la vida de las personas afectadas (Nurmi & Tarkoma, 2017). Esto acontecimientos deben ser acompañados con respuestas rápidas que permitan reducir los daños tanto humanos como materiales, para así evitar una disrupción en el funcionamiento de la sociedad. Las operaciones de búsqueda y rescate son un medio por el cual se puede presentar esta respuesta veloz y efectiva (Goverment of Canada, 2018), donde las operaciones más comunes son las de tipo urbano.

A lo largo de los años diversos avances tecnológicos han sido implementados en el uso de estas operaciones de búsqueda y rescate urbanos. Siendo uno de estos elementos el uso de robots. Los robots en este tipo de operaciones han resultado de gran ayuda, ya que han permitido mejorar tiempos de respuesta y ofrecen a rescatistas un mayor entendimiento de la situación (Cubber et al., 2017) , tales como dificultad de control, comunicación e integración entre estos y sus controladores. Esfuerzos se han realizado para ayudar a estos inconvenientes, entre estos la implementación en tecnologías de la nube (Mouradian et al., 2018). Mientras esto resulta prometedor, la manera en que se lo ha realizado aun requiere de implementaciones complejas y alta especialidad (Mouradian et al., 2018).

De esta manera, esta investigación propone realizar una arquitectura que se enfoque en el control y monitoreo de robots de búsqueda y rescate, que este basado en tecnologías de la nube, pero en este caso se enfoque en arquitecturas de tipo serverless.

De esta manera seria posible reducir las complejidades de implementación y uso de soluciones basadas en la nube, además de reducir la necesidad de manejar infraestructura tanto física como virtualizada.

El desarrollo de esta investigación resultaría en una arquitectura de control y monitoreo que sería efectiva en operaciones de búsqueda y rescate urbanos, que otorgaría a los robots la capacidad de responder de una manera más rápida y ser controlados y monitoreados de una forma sencilla y versátil.

1.5 ALCANCE

Esta investigación busca evaluar que una arquitectura de control y monitoreo de robots para búsqueda y rescate urbanos basados en tecnologías serverless en la nube reduciría la complejidad de implementación y operación a un costo reducido de estos robots. Para lograr realizar esto se limitará a desarrollar un prototipo que aplique esta arquitectura y a base de distintas pruebas se evaluará el comportamiento de la arquitectura. Con esto, se realizarán comparaciones con las necesidades actuales que requieren los robots de búsqueda y rescate urbanos. Así, se determinará la utilidad de la arquitectura y sus limitaciones, así como los posibles trabajos que podrán surgir del desarrollo de esta investigación.

Para delinear de forma adecuada el alcance de la investigación planteada, se proponen varias preguntas de investigación asociadas a los objetivos específicos, tal como se muestra en la Tabla 1.

Tabla 1

Preguntas de Investigación

Objetivo específico		Pregunta de investigación	
i.	**Investigar herramientas, metodologías y modelos utilizados en las operaciones de búsqueda y rescate urbanos para poder relacionarlas con las capacidades de las herramientas de la nube.**	a.	¿Cómo se lleva a cabo de manera formal una operación de búsqueda y rescate urbanos con el uso de robots?
		b.	¿Qué elementos son necesarios para una operación de búsqueda y rescate urbanos?
ii.	**Diseñar una arquitectura que permita el control y monitoreo de diversos robots de manera simultánea por medio de herramientas basadas en la nube y de tipo serverless.**	c.	¿Qué elementos debe de tener una arquitectura de control y monitoreo para que sea útil en una operación de búsqueda y rescate urbanos?
		d.	¿Qué herramientas se pueden aplicar a la arquitectura de tal manera que reduzca complejidad de implementación?
		e.	¿Qué características deben tener él robot para que pueda funcionar en conjunto con la arquitectura de control y monitoreo planteado?

Continua

iii.	**Implementar la arquitectura de control y monitoreo con un prototipo en un entorno controlado de tal manera que se pueda medir su funcionamiento y utilidad.**	f. ¿Qué facilidad se presenta al desplegar el modelo planteado? g. ¿Qué características deberá tener el entorno controlado para obtener resultados confiables?
iv.	**Validar la arquitectura de control y monitoreo con pruebas de QoS, para así poder determinar sus limitaciones y su nivel de utilidad en una operación de búsqueda y rescate urbano.**	h. ¿Qué rendimiento, velocidad, latencia, escalabilidad y disponibilidad tiene la arquitectura? i. ¿Qué utilidad tendrá la arquitectura en una operación de búsqueda y rescate urbanos? j. ¿Cuáles son las limitaciones de la arquitectura planteada? k. ¿Cuáles serían los posibles trabajos futuros que podrán surgir de la investigación?

1.6 DEFINICIÓN DE LA INVESTIGACIÓN

Para llevar a cabo la investigación se ha establecido una metodología específica para este proyecto que está basado en diferentes fases en las que se busca un entendimiento completo de la problemática y así llevara a la mejor solución a base del objetivo propuesto. Se han definido las siguientes fases concretas:

a. FASE I - Investigación: Durante esta fase lo que se realiza es estudiar la problemática en toda su profundidad. En este caso, esto se refiere al estudio de las operaciones de búsqueda y rescata urbana y el uso de robots en este tipo de operaciones. Se debe determinar cómo funcionan estos robots, que requerimientos tienen para ser objetivos, como se evalúa su funcionalidad y como se operan. De esta forma, a su vez, también se debe investigar sobre como las herramientas basadas en la nube son capaces de ser implementadas en robots de búsqueda y rescate urbano.

b. FASE II - Diseño: Con los resultados de la primera fase se lleva a cabo un diseño que pueda cumplir con el objetivo de todo el proyecto. Para esto es necesario explorar y analizar herramientas que sean necesarias para la arquitectura, sus formas de comunicación y los requisitos en sí de la arquitectura de control y monitoreo de robots de búsqueda y rescate urbanos.

c. FASE III - Implementación: Con un diseño realizado, es necesario implementarlo en un ambiente real. Se aplica cada una de las especificaciones dadas en el diseño y se establece un entorno donde este podrá ser probado de una forma controlada y adecuada.

d. FASE IV - Validación: Se realizan pruebas a la arquitectura que fue diseñado e implementado. Estas pruebas deben validar la usabilidad de la arquitectura y su aplicación en operaciones de búsqueda y rescate. En esta fase se hace uso de todos los elementos de las fases anteriores para medir y comparar desempeño, funcionalidad, aplicabilidad y limitaciones. Esta fase permite llegar a conclusiones y recomendaciones.

CAPÍTULO II

MARCO TEÓRICO Y ESTADO DEL ARTE

2.1 MARCO TEÓRICO

2.1.1 Planteamiento del marco teórico

2.1.1.2 Categorización de temáticas

Conforme con la naturaliza de la investigación se ha identificado que existen 2 grandes temáticas que son necesarias a explorar: Las operaciones de búsqueda y rescate, y el cloud computing.

La categorización de temáticas permite identificar claramente cuáles son los temas que deben ser desarrolladas y llevadas a cabo en el marco teórico, para los elementos más importantes de la investigación. Gracias a esta categorización se pueden tener temas a desarrollar que sean relevantes tanto para las soluciones propuestas como para el problema planteado.

Para esta categorización se ha decido establecer 4 categorías de variables, yendo desde el campo más general hasta el más específico.

Para la exploración del cloud computing se tiene los siguientes niveles:

- Nivel 1: Cloud Computing.
    - Nivel 2: Serverless Computing
        - Nivel 3: Cloud Robotics
            - Nivel 4: Arquitectura de control y Monitoreo de robots en operaciones de búsqueda y rescate urbano.

Para la exploración de operaciones de búsqueda y rescate se tienen los siguientes niveles:

- Nivel 1: Búsqueda y Rescate Urbano
    - Nivel 2: Operaciones de Búsqueda y Rescate Urbano
        - Nivel 3: Robots de búsqueda y rescate urbano
            - Nivel 4: Costos, y complejidad técnica y operativa en el manejo de robots en operaciones de búsqueda y rescate urbanos.

2.1.2 Cloud computing

Cloud computing es un término que representa la provisión de recursos computacionales a través de la internet por medio de servicios medidos, provistos por alguna empresa (Rittinghouse & Ransome, 2017). Es una tecnología que ya lleva algún tiempo en desarrollo, pero ha sido en la última década que a presenta un mayor auge y desarrollo, a medida que más y más empresas han empezado hacer uso de este concepto. Cloud computing viene siendo una evolución de diferentes conceptos computacionales originados a finales del siglo XX, tales como grid computing y clúster computing. Cloud computing hace uso de estas tecnologías para la adecuada provisión de sus servicios a través del internet (Rittinghouse & Ransome, 2017).

El funcionamiento del cloud computing surge como una división de su arquitectura en diferentes partes a base de los servicios que ofrece un proveedor de cloud. Esta arquitectura surge de infraestructura física y la virtualización de recursos computacionales. A partir de allí, se forman 3 elementos: IaaS, PaaS y SaaS (Sadiku,

Musa, & Momoh, 2014). Estos elementos de arquitectura son los que son eventualmente provistos a usuarios finales y formar el modelo de negocio para las empresas proveedoras de cloud computing (Sadiku et al., 2014).

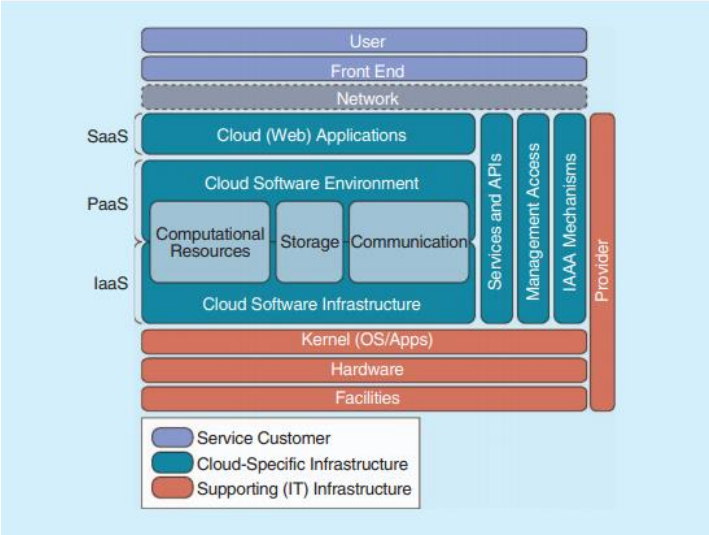

Figura 1. Arquitectura generalizada de Cloud Computing

Fuente: (Sadiku et al., 2014)

A base de la **figura 1,** se pueden apreciar los distintos modelos existentes dentro del Cloud Computing.

IaaS que por sus siglas en ingles significa Infraestructura as a Service, que en español es Infraestructura como servicio es una de las ofertas más simples dentro del cloud computing. Consiste en la entrega de recursos de computación como almacenamiento, procesamiento y red en el nivel más bajo de abstracción posible. Este modelo da un amplio nivel de control a los usuarios sobre cómo se hace uso de los recursos otorgados. Estos recursos computacionales son accedidos remotamente y su

costo se da por uso. Generalmente funciona gracias a la virtualización (Sadiku et al., 2014).

PaaS, que por sus siglas en ingles significa Platform as a Service, que en español es Plataforma como servicio es el siguiente nivel de entrega de servicios en la nube. Lo que permite es que diferentes aplicaciones generen una interfaz con aplicaciones cloud. Lo que se presenta es una plataforma desde la cual se puede trabajar para desarrollar aplicaciones web. Se pueden usar diversos recursos de la nube, pero el control se limita a lo entregado por la plataforma. Este modelo de entrega de servicio quita control al usuario, pero facilita el desarrollo de aplicaciones (Sadiku et al., 2014).

SaaS, que por sus siglas en ingles significa Software as a Service, que en español es Software como servicio, es el último nivel en la entrega de servicios en la nube. Este nivel integra software completo que se lo entrega al usuario a través de la nube. Este software es entregado a través del internet y el cliente no tiene mayor control sobre él, más que uso. Permite a empresa evitar la instalación de software, manejándolo desde el internet (Sadiku et al., 2014).

La entrega de estos diferentes modelos y tipos de servicios encontrados en la nube lo realizan una variedad de proveedores (Rittinghouse & Ransome, 2017). En la actualidad existe una cantidad limitada de proveedores de servicios de la nube a nivel mundial, algunos de los más destacados son (Rittinghouse & Ransome, 2017; Sadiku et al., 2014) :

- Amazon Web Services

- Salesforce
- Microsoft Azure
- Google Cloud Platform

2.1.3 Serverless Computing

Serverless computing es un paradigma dentro del campo de cloud computing que busca eliminar la necesidad de que el usuario haga uso o gestione un servidor para sus aplicaciones y a su vez facilite el desarrollo de aplicaciones en el internet. Actualmente serverless computing funciona a base de funciones sin estado y eventos (McGrath & Brenner, 2017). Esto quiere decir que su uso basa en que aplicaciones están definidas por acciones, y estas acciones ejecutan funciones definidas a base de una arquitectura serverless (McGrath & Brenner, 2017). Todas estas funciones son definidas por código provisto por el desarrollo y no existe un control directo de hardware sobre el que corren estas funciones, permitiendo que estas sean auto escalables y elásticas fácilmente (McGrath & Brenner, 2017). Debido al funcionamiento de serverless computing, se lo puede confundir con propuestas PaaS, pero la principal diferencia radica que una solución serverless se basa en la utilización de funciones para las aplicaciones y no existe en si un entorno donde se despliega toda la aplicación (Baldini et al., 2017). Se puede ver el nivel sobre el que trabaja serverless en la **figura 2**.

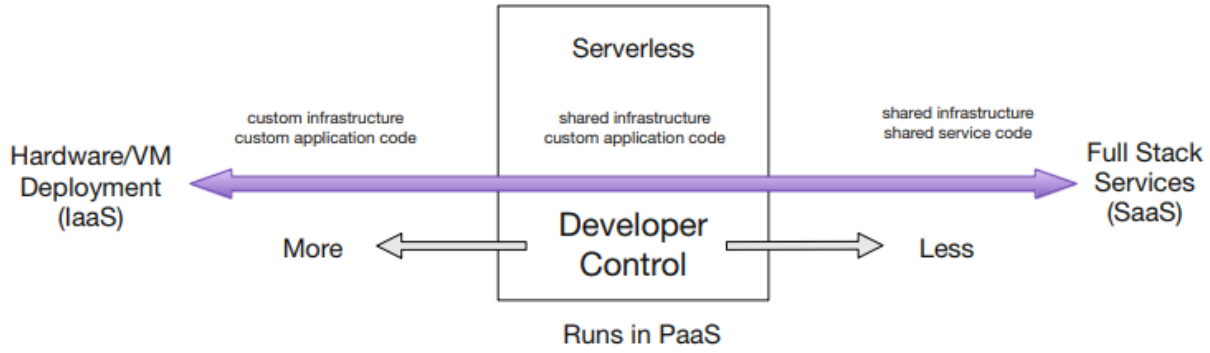

Figura 2. Ubicación y función de serverless dentro de los distintos modelos de cloud computing.

Fuente: (Baldini et al., 2017)

A base de la **figura 2**, se entiende que serverless opera en PaaS, es decir que la infraestructura no se administra y tan solo se maneja código. Pero esto no quiere decir que serverless sea lo mismo que PaaS.

El funcionamiento y arquitectura de serverless, que se lo puede observar en la **figura 3**, se basa en correr una función cuando existe una petición de esta, registrar su uso, medirlo y cuando ya se complete esa función para la petición dada, terminar la petición (Baldini et al., 2017). Todo esto se lo realiza dinámicamente y con la ayuda de servidores que se instancian o terminan dependiendo de la cantidad de peticiones existente. Serverless hace uso de servidores, pero no se puede saber cuáles son estos o tener control sobre ellos. El uso de serverless se combina con otros servicios que no requieren gestión de servidores (Baldini et al., 2017).

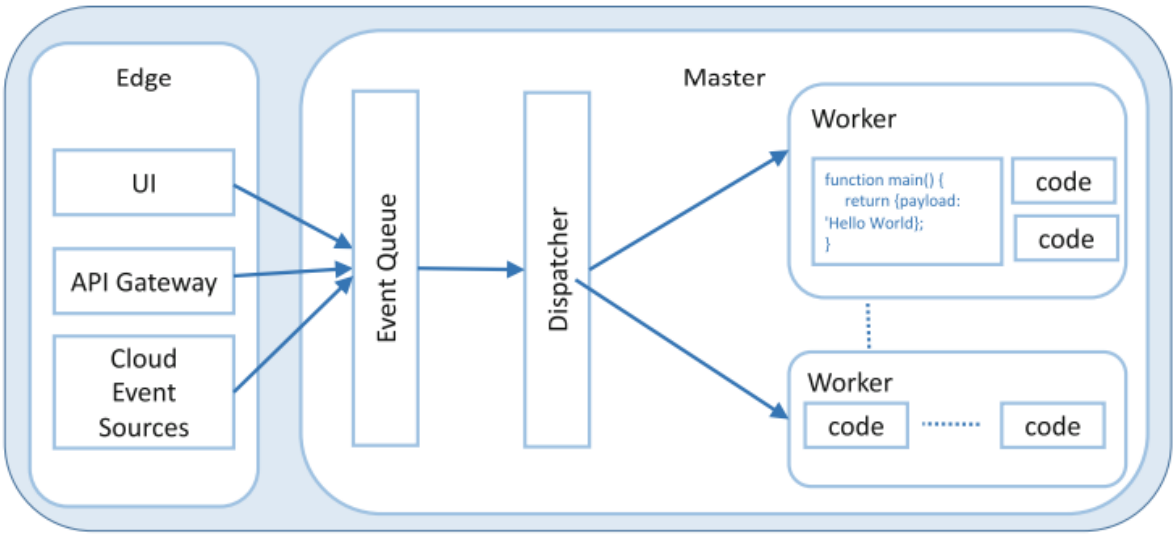

Figura 3. esquema de funcionamiento de serverless computing

Fuente: (Baldini et al., 2017)

Actualmente los principales proveedores de soluciones serverless son AWS con AWS Lambda, Google con Google Cloud Functions y Firebase, Microsoft Azure Functions e IBM OpenWisk (McGrath & Brenner, 2017).

2.1.4 Cloud Robotics

Cloud Robotics, como su nombre lo indica es la combinación del uso de cloud computing en conjunto con la robótica. Cloud robotics, permite que robots puedan delegar muchas de sus funciones a servicios externos en la nube, de esta forma logrando sistemas complejos y de alta funcionalidad que no serían posible por medios tradicionales (Wan et al., 2016).

La manera en que funciona esta tecnología es por medio de la implementación de una red de robots conectados a la nube. Esto se logra por medio de una arquitectura que

presenta dos elementos: La plataforma cloud y los robots. Los robots pueden ser de cualquier tipo, solo debe que conectarse la plataforma cloud. La plataforma cloud está compuesta por servidores, bases de datos, funciones, etc (Wan et al., 2016).

Gracias a este concepto, surgen de diversas aplicaciones como por ejemplo la gestión de logística, análisis de datos, operaciones coordinadas entre robots y también propuestas del uso de estas tecnologías dentro de operaciones de búsqueda y rescate (Wan et al., 2016).

2.1.5 Arquitecturas de control y monitoreo de robots en la nube

El control y monitor de robots, es un elemento que se ha usado desde la existencia de los primeros robots y es un elemento esencial para mantener y hacer uso de estos robots. Una de las propuestas que existen para controlar y monitorear estos robots es el uso de la nube y más específicamente cloud robotics (Nakamura et al., 2018). Esto permite un control y uso de recursos masivo que no sería posible de manera tradicional. Se puede apreciar una arquitectura para el control de estos robots en la **figura 4**.

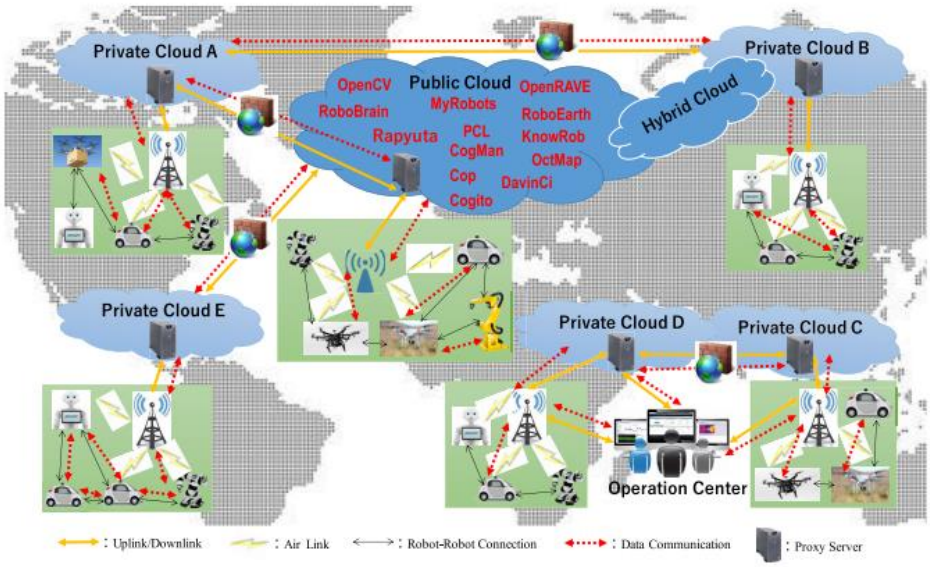

Figura 4. Esquema de funcionamiento de cloud robotics

Fuente: (Nakamura et al., 2018)

La figura 4, muestra el funcionamiento de robots por medio de cloud robotics. Los diferentes robots, sin importar su tipo ni aplicación, se conectan a través de internet a diferentes servicios de nube o a través de una red a una nube privada. La nube, provee a estos robots diferentes servicios sea para procesamiento de datos, control o monitoreo.

El monitoreo y control funciona a base de las conexiones existentes de los robots y la nube. Un robot puede enviar información a través de la nube y a su vez recibir comunicación, logrando no solo que este sea controlado sino también una comunicación con otros robots (Nakamura et al., 2018).

2.1.6 Búsqueda y rescate urbano

La búsqueda y rescate es un término que describe el proceso de la búsqueda y provisionamiento de ayuda a personas que se encuentran en peligro, con el propósito de

reducir daños y salvar vidas (ICARUS, n.d.); la búsqueda y rescate es un campo muy gran de acción y presenta muchos tipos basados en donde y como se aplican; la búsqueda y rescate urbano es uno de estos campos.

La búsqueda y rescate urbano es un tipo de búsqueda y rescate que se enfoca primordialmente la localización, extracción, y la asistencia médica inmediata de víctimas que se encuentran atrapados en espacios confinados debido a desastres naturales, danos estructurales, accidente o colapsos (ICARUS, n.d.).

Las operaciones para llevar a cabo búsqueda y rescate urbano usualmente son llevadas a cabo por equipos especializados en cada país y generalmente lo hacen con la ayuda de guías y metodologías presentadas por el UN's International Search and Rescue Advisory Group (INSARAG) (INSARAG, n.d.-c).

2.1.7 Operaciones de búsqueda y rescate urbano

Las operaciones de búsqueda y rescate se refieren al proceso de llevar a cabo la búsqueda y rescate. Cada país puede establecer sus propios procedimiento y normas para llevar a cabo este tipo de operaciones, pero esto no quiere decir que existan guías generales y recomendaciones (INSARAG, n.d.-a). INSARAG, un grupo de la ONU presenta diferentes guías que ayudan a equipos establecer operaciones de búsqueda y rescate de tal manera que pueda escalar y acoplarse con operaciones en otros países y que puedan ser usados tanto para operaciones de gran escala o de pequeña escala (INSARAG, n.d.-c).

Una operación de búsqueda y rescate presenta en general 5 fases principales que se aprecian en la **figura 5** con el manejo de un ciclo a lo largo de la operación.

Figura 5. Fases de desarrollo de operaciones de búsqueda y rescate según INSARAG

Fuente: (INSARAG, n.d.-a)

Las fases presentadas en la **figura 5** son:

- **Preparación:** Es la fase previa a la respuesta ante un desastre y es donde los equipos se preparan a sí mismo para la operación y los diferentes equipos que se llevaran (INSARAG, n.d.-a).
- **Movilización:** Es la fase donde los equipos se movilizan a la zona de desastre. (INSARAG, n.d.-a)

- **Operaciones:** Son todas las acciones que se realizan desde que el equipo de búsqueda y rescate llega a la zona de desastre hasta que termina y sale de la zona de desastre (INSARAG, n.d.-a).
- **Desmovilización:** Es la fase donde el equipo de búsqueda y rescate tiene que salir de la zona y terminar la operación (INSARAG, n.d.-a).
- **Post Misión:** Es el regreso del equipo a su base y donde se realiza un informe de toda la operación (INSARAG, n.d.-a).

2.1.8 Robots de búsqueda y rescate urbano

La búsqueda y rescate y más específicamente la búsqueda y rescate urbano, es un campo donde la aplicación de la tecnología es esencial y permite tener operaciones más efectivas. Una de estas tecnologías es la implementación de robots (Cubber et al., 2017).

Los robots de búsqueda y rescate urbano son un tipo de robots que están diseñados e implementan elementos que ayudan a equipos humanos a llevar actividades que permitan localizar, monitorias y rescatar personas u objetos en sitios de riesgo (Cubber et al., 2017). Generalmente estos robots son altamente variados y se los puede encontrar tanto como vehículos aéreos o terrestres en una amplia variedad de tamaños. Actualmente podemos encontrar robots que realizan las siguientes actividades (Murphy, Tadokoro, & Kleiner, 2016) :

- Búsqueda
- Mapeo y Reconocimiento
- Limpieza de escombros

- Inspección Estructural
- Asistencia médica en sitio
- Evacuación de victimas
- Remplazo de operadores humanos
- Soporte logístico

Para lograr esto, se designa estas actividades a diferentes tipos de robots como son (Murphy et al., 2016):

- UGV o Unmanned Ground Vehicles, que pueden ser tanto de grandes dimensiones, como sumamente pequeños.
- UAV o Unmanned Aerial Vehicles.
- UMV o Unmanned Marine Vehicles.
- UUV o Unmanned Underwater Vehicles.
- USV o Unmanned Surface Vehicles.

2.1.9 Costos, complejidad técnica y operativa en manejo de robots de búsqueda y rescate urbano

El manejo y uso de robots de búsqueda y rescate urbano ha mostrado un claro beneficio durante este tipo de operaciones (Cubber et al., 2017). Pero, aun así, se presentan diversas dificultados todavía. Los retos y complicaciones que se tienen en la actualidad son (Cubber et al., 2017; Murphy et al., 2016):

- Despliegue lento de los robots de búsqueda y rescate y sus herramientas complementarias.

- Autonomía limitados desde los puntos de vista de manejo de energía e inteligencia o capacidad de procesamiento de los robots.
- Nivel de colaboración entre los robots es limitada, dificultando operaciones coordinadas.
- Integración de robots con herramientas altamente especializado es altamente limitado.
- Complejidad de aprendizaje y entrenamiento para el uso de robots de búsqueda y rescate.
- Problemas de interoperabilidad entre los equipos de búsqueda y rescate y los robots.

2.2 ESTADO DEL ARTE

El análisis de estado de arte acerca del uso de modelo de arquitectura basadas en la nube para su uso en operaciones de búsqueda y rescate, se llevó a cabo una revisión de literatura inicial que esta baso en las guías propuestas por (Kitchenham & Charters, 2007). Se siguió las siguientes fases:

2.2.1 Planteamiento de la revisión de literatura

Lo que se llegó a realizar es una descripción de la problemática de investigación, para así obtener un contexto claro para la búsqueda de investigaciones científicas. De allí, se presentó el objetivo de búsqueda en conjunto con diversas preguntas de investigación. Así, se pudo alinear la búsqueda al problema de investigación y por último definir criterio para la inclusión y exclusión.

2.2.2 Conformación del grupo de control y extracción de palabras relevantes para la investigación

A base de (Petersen, K. and Feldt, R. and Mujtaba, S. and Mattsson, 2008) es esencial establecer artículos que son relevantes para la investigación, quitando así los que no abarcan completamente el enfoque principal. Con esto, se llevó a cabo la selección de un grupo de control que resulta un punto de partida para el resto de la revisión de literatura.

Se hizo uso de 2 investigadores, y se llegó a un acuerdo de presentar diversos estudios como candidatos para el grupo de control. Con esto se llegó a tener un grupo de control que se presenta en la Tabla 1.

Tabla 2

Grupo de Control

Título	**Cita**	**Palabras Clave**
Robots as-a-service in cloud computing: Search and rescue in large-scale disasters case study	(Mouradian et al., 2018)	Cloud Computing, Infrastructure as a Service, Internet of Things, Robot as-a-Service, **Search and Rescue, Cloud Robotics.**

El grupo de estudio, en este caso es bastante limitado debido a que existe una carencia de estudio en el campo, por ser un campo bastante nuevo.

2.2.3 Construcción y afinación de cadenas de búsqueda

En primer lugar, se estableció la búsqueda de artículos que tan solo estén relacionados con el problema a explorar, pero resulto que este es amplio y existe una gran cantidad de estudios relacionados en una amplia variedad de campos. Esto en principal debido a la gran variedad de formas a la que se puede abarcar la robótica. Por esta razón, se decidió delimitar la investigación al campo de la computación en la nube y el internet de las cosas, siendo estos elementos relacionados con la solución planteada. Estos campos al ser bastante nuevos son menos explorados en el campo de la robótica para la búsqueda y rescate y por lo tanto se tiene una cantidad de investigaciones medibles y de gran aporte para la investigación.

Con las palabras clave del grupo de control, se estableció una cadena de búsqueda que abarcara estas palabras clave. La cadena de búsqueda quedo de la siguiente manera: (SEARCH) AND (RESCUE) AND ((CLOUD COMPUTING) OR (CLOUD ROBOTICS) OR (INTERNET OF THINGS)), la cadena de búsqueda resulta dar artículos no relacionados con el tema de investigación, por lo cual se agregó términos relacionados a robots. La nueva cadena de búsqueda resulto ser: (((SEARCH) AND (RESCUE)) AND ((ROBOT) OR (ROBOTS) OR (DRONES)) AND ((CLOUD COMPUTING) OR (CLOUD ROBOTICS) OR (INTERNET OF THINGS))).

Al probar la cadena de búsqueda presento datos limitados, por lo tanto, se decido establecer esta cadena como definitiva.

2.2.4 Selección de estudios

La cadena se le aplico a la base digital IEEE Explore. Se obtuvieron un total de 37 artículos que presentaban relación con el tema propuesto. Además, el grupo de control apareció como un resultado con la cadena de búsqueda seleccionado.

De la cantidad seleccionada se estableció los siguientes filtros:

- **Fecha de publicación**: Solo analizar artículos que hayan sido publicados desde 2014 a 2019. Esto se lo realiza debido al rápido avance de la tecnología donde estudio antiguos se vuelven obsoletos y soluciones más modernas existen.
- **Tipo de artículo**: Se acepto tan solo artículos de conferencia y de revistas.
- **Forma de búsqueda**: Solo se realizó búsqueda de artículos por su meta data.

Además de estos filtros, se procedió a revisar cada uno de los artículos y sus temáticas, haciendo énfasis en su abstract y conclusión. A base de todo esto, se eligieron 10 artículos representados en la tabla 2.

Tabla 3

Artículos presentados para estado del arte.

Código	Titulo	Cita
EP1	Robots as-a-service in cloud computing: Search and rescue in large-scale disasters case study	(Mouradian et al., 2018)

Continua

EP2	A Study of Robotic Cooperation in Cloud Robotics: Architecture and Challenges	(Nakamura et al., 2018)
EP3	Training and Support system in the Cloud for improving the situational awareness in Search and Rescue (SAR) operations	(Pelka et al., 2014)
EP4	Design and Implementation of Debris Search and Rescue Robot System Based on Internet of Things	(Xin, Qiao, Hongjie, Chunhe, & Haikuan, 2018)
EP5	IoT Sensor Based Mobility Performance Test-Bed for Disaster Response Robots	(Kim, Jung, Gu, Kim, & Song, 2017)
EP6	Autonomous cloud-based drone system for disaster response and mitigation.	(Alex & Vijaychandra, 2017)
EP7	Real-Time, Cloud-Based Object Detection for Unmanned Aerial Vehicles	(Lee, Wang, Crandall, Sabanovic, & Fox, 2017)
EP8	Internet of Drones	(Gharibi, Boutaba, & Waslander, 2016)

Continua

EP9	Collab-SAR: A Collaborative Avalanche Search-and-Rescue Missions Exploiting Hostile Alpine Networks	(Rahman, Azad, Asyhari, Bhuiyan, & Anwar, 2018)
EP10	Semantic data exchange between collaborative robots in fog environment: ¿Can CoAP be a choice?	(Dey, Bhattacharyya, & Mukherjee, 2017)

2.2.5 Elaboración del estado del arte

EP1 (Mouradian et al., 2018) **: Robots as-a-service in cloud computing: Search and rescue in large-scale disasters case study**

Se presenta un modelo para la entrega de robots de búsqueda y rescate como un servicio basado en la nube. Para esto, propone una arquitectura que presenta la capacidad de tener modelos de robots en conjunto con sus características y también un mercado para los mismos. Todo esto lo hace por medio de la implementación de controles con el uso de Internet de las Cosas e Infraestructure as a Service. Este artículo presenta que existen dificultades en la homogenización de los robots y sus recursos de IoT. También presenta que existe altos riesgos de la degradación de la calidad de servicios de estos robots, en especial en su control debido a la gran variabilidad que se presenta.

EP2 (Nakamura et al., 2018) **: A Study of Robotic Cooperation in Cloud Robotics: Architecture and Challenges**

Se realiza un análisis extenso sobre la aplicación de la nube en la robótica, sus principales retos y posibilidades. Aunque no se centra específicamente en operaciones de búsqueda y rescate, hace uso a este tipo de operaciones como un ejemplo principal para el uso de estos robots. A lo largo del paper se presenta la capacidad de hacer uso de la nube para delegar capacidad de procesamiento a los robots, implementar análisis de big data, inteligencia artificial, etc. A su vez, hace notar que en la actualidad se presentan retos como: la calidad de servicios, rendimiento y problemas de seguridad que pueden limitar el uso de la nube en la robótica. Aun así, considera que el uso de nube en la robótica es un elemento esencial que sería de gran utilidad en todo tipo de aplicaciones, en especial en operaciones de búsqueda y rescate.

EP3 (Pelka et al., 2014) **: Training and Support system in the Cloud for improving the situational awareness in Search and Rescue (SAR) operations**

Se presenta un Sistema de entrenamiento y soporte para el proyecto ICARUS, un Proyecto de robótica para operaciones de búsqueda y rescate. El Sistema presenta la capacidad de aumentar el entendimiento de la situación de riesgo con la generación de mapas en 3D. Esto se lo realiza con el uso de robots y servicios en la nube, donde los robots al enviar datos del terreno a la nube generan la capacidad de crear extensos análisis que dan resultado mapas que ayudan a equipos de búsqueda y rescate mejorar su campo de acción. Se presenta como esta forma de monitoreo y análisis puede ser usada tanto en robots terrestres como aéreos.

EP4 (Xin et al., 2018) **: Design and Implementation of Debris Search and Rescue Robot System Based on Internet of Things**

Se presenta un prototipo de un simple robot para operaciones de búsqueda y rescate que es controlado a través de una red de internet WiFi por una terminal de control. Se presenta la capacidad de controlar movimiento y monitorear tanto ubicación como estado del robot a través de la red. Aunque presenta un control novedoso, solo se lo realiza por medio de una red WiFi lo que genera una limitación de rango de acción y capacidad de uso.

EP5 (Kim et al., 2017)**: IoT Sensor Based Mobility Performance Test-Bed for Disaster Response Robots**

El articulo presenta una aplicación del internet de las cosas en el campo de búsqueda y rescate para el monitoreo de robots. Pero, en lugar de presentar un monitoreo en campo, lo hace en entornos de entrenamiento para evaluar y mejorar los robots. El artículo propone la construcción de diferentes pistas de entrenamiento que en conjunto de diversos sensores en robots y en la pista, se pueda ver como este se comporta y funciona. Toda la información que se recolecta del entrenamiento se envía por internet, se recolecta y se analiza. Así se tiene la capacidad de ajustar al robot para que sea más eficiente o ajustar la pista para realizar futuras pruebas.

EP6 (Alex & Vijaychandra, 2017)**: Autonomous cloud-based drone system for disaster response and mitigation.**

En el artículo, se presenta un sistema que implementa una red compleja de comunicación entre drones y la nube en casos de búsqueda y rescate. El sistema presenta una plataforma de control que permite que drones avanzados recolecten datos del terreno que es almacenada y procesada en la nube para crear mapas de sitio. Tal información es dada a drones más simples y menos costos. Así, una gran variedad de drones se vuelve capaces de navegar una zona sin necesidad de tener sensores complejos, reduciendo costes de la operación de búsqueda y rescate y aumentando cantidad de drones posibles a desplegar.

EP7 (Lee et al., 2017)**: Real-Time, Cloud-Based Object Detection for Unmanned Aerial Vehicles**

El artículo, propone una arquitectura basada en la nube por la cual drones pueden usarla para hacer uso de redes neuronales complejas para la detección de objetos en el espacio. De esta forma se establece la reducción de complejidad requerida para implementar drones inteligentes en operaciones de búsqueda y rescate. La propuesta hace uso de drones de bajo costo y redes neuronales de tipo CNN (Red Neuronal Convolucional). Esto permitió la detección exitosa en una amplia variedad de objetos al mismo tiempo y en tiempo real. El reto se presenta en mantener una conexión con poca latencia para asegurar la efectividad del análisis de imágenes.

EP8 (Gharibi et al., 2016)**: Internet of Drones**

El articulo propone una arquitectura de control y monitoreo de drones aéreos al que se lo establece con el nombre de “Internet of Drones”. La arquitectura presenta una

estructura que permite comunicación, definición de espacios aéreos y operación de drones a través de internet. Se divide a la arquitectura en una cantidad de capas lógicas que permite a los drones operar de distintas formas a diferentes niveles de aplicabilidad. Tanto para vuelo, comunicación o control. En si el articulo presenta una novedosa arquitectura que podría permitir a diferentes drones operar en un mismo entorno de manera coordinada, sencilla y transparente, permitiendo así su implementación en diferentes campos donde sea requerido el uso de robots aéreos.

EP9 (Rahman et al., 2018)**: Collab-SAR: A Collaborative Avalanche Search-and-Rescue Missions Exploiting Hostile Alpine Networks**

El artículo, propone una arquitectura de comunicación entre drones, equipos y humanos que permite el desarrollo de una operación de búsqueda y rescate colaborativa en los Alpes con el uso de protocolos y conexiones asociadas a internet. La arquitectura propuesta resulta efectiva para que drones sean capaces de enviar información a equipos humanos entre diferentes equipos de búsqueda y rescate en un amplio radio de operación. Así, la arquitectura permite que los equipos se comuniquen de una forma efectiva en una operación de búsqueda y recates. Permitiendo resultados exitosos y rápidos.

EP10 (Dey et al., 2017)**: Semantic data exchange between collaborative robots in fog environment: Can CoAP be a choice?**

El artículo, explora la utilización de CoAP (Constrained Application Layer Protocol) como una forma de intercambio de datos entre robots de búsqueda y rescate y

dispositivos de borde que se conectan a la nube. Con este tipo de conexión y comunicación, se pretende una mayor implementación de conceptos como Internet de las cosas y Cloud Computing dentro de la robótica relacionada con la búsqueda y rescate. El artículo da como resultado que el uso de protocolos como CoAP, en conjunto con estructura de comunicación robustas para los robots, puede ser prometedor para una comunicación con infraestructura de borde y conexiones hacia internet.

2.2.6 Característica del estado del arte

Los trabajos presentan un amplio rango de acercamientos y soluciones a implementar en la robótica para operaciones de búsqueda y rescate. Las soluciones generalmente se enfocan en mejorar las capacidades de los robots de búsqueda y rescate, y a su vez reducir sus costos de operación. Las soluciones son variadas, pero todas carecen de una integración completa para una operación de búsqueda y rescate y se enfocan en elementos muy específicos como mapeo, monitoreo o control. Las soluciones que busquen un enfoque de aplicación más amplio presentan arquitecturas complejas de implementar, aunque con un gran potencial. También, estas en su mayoría se enfocan más en la arquitectura de conexión a internet, en lugar del uso de la nube como una herramienta completa. Aun así, los estudios demuestran la capacidad y potencial que tiene el uso de elementos como la nube y el internet de las cosas en robots de búsqueda y rescate, aunque también destacan retos como lo que es la latencia y rendimiento de conexión. Además de esto, se toma de los trabajos para esta investigación el uso del cloud computing para la comunicación y monitoreo, el uso de WiFi como medio de conexión y la capacidad de manejar una cantidad indefinida de robots.

CAPÍTULO III

DISEÑO DE LA ARQUITECTURA

3.1 INTRODUCCIÓN

El desarrollo de este proyecto se basa en estructurar una arquitectura de control y monitoreo de robots de búsqueda y rescate urbano por medio del uso de hardware, software y tecnologías serverless basadas en la nube y para eso es necesario tomar en cuenta una serie de características funcionales:

- Control: Se refiere a la capacidad de un individuo de comandar a uno o varios robots desde una parte remota conectado a un entorno de control. El control implica movimiento y la activación de componentes.
- Monitoreo: Se refiere a la capacidad que tiene un individuo de visualizar el estado de robots que estén activos a través de la nube. Este estado incluye diferentes variables tales como posicionamiento, movimiento y otros datos obtenidos de sensores.
- Robots de búsqueda y rescate: Son aquellos dispositivos controlados de manera remota que se registran a la arquitectura para su control y monitoreo. Estos robots deben tener la capacidad de ser comandados y de enviar sus datos de sensores y componentes a través de la nube. Cualquier robot que pueda mantener una conexión a internet y adherirse a la arquitectura puede ser comandado y monitoreado.
- Tecnologías serverless de la nube: Estas tecnologías se encuentran en variedad de formas. Se tiene ejemplos como AWS Lambda o Google Cloud Functions que

se basa en el despliegue de funciones stateless a través de endpoints. Además de estos se tienen plataformas que presentan soluciones para almacenamientos y despliegue sin servidor como Firebase.

Con todo esto, además, se logró que la arquitectura permita cumplir los procesos de una operación de búsqueda y rescate de una manera completa. Estas características, siguen el modelo de la INSARAG para el proceso de búsqueda y rescate que presenta las siguientes fases: Preparación, Movilización, Operaciones, Desmovilización y Post Misión. (INSARAG, n.d.-a)

3.2 HERRAMIENTAS

Las herramientas elegidas para poder llevar a cabo objetivos establecidos fueron tales que permitieron una comunicación eficaz, flexibilidad, adaptabilidad y facilidad de uso. Las herramientas las podemos encontrar en 4 categorías: Hardware, Software, Lenguajes de Programación y Herramientas de la Nube.

3.2.1 Hardware

3.2.1.1 Raspberry Pi 3

Es una computadora de bajo costo de reducidas dimensiones que permite realizar una amplia variedad de acciones equivalentes a lo que puede realizar un computador normal. Además, tiene la capacidad de conectarse y comandar dispositivos externos como sensores, motores, etc (RASPBERRY PI, n.d.-e).

Está construida a base de una arquitectura ARM y por lo tanto solo permite usar sistemas operativos que soportes ARM, por lo general Linux. Raspbian, una distribución

de Linux, es el sistema operativo principal usado por la Raspberry, aunque existen alternativas diferentes (RASPBERRY PI, n.d.-c).

Se usa en diferentes campos; principalmente la educación y el prototipado, aunque también se la puede encontrar en la industria (RASPBERRY PI, n.d.-e).

Figura 6. Raspberry Pi 3 Moldelo B+

Fuente: (RASPBERRY PI, n.d.-a)

3.2.2 Software

3.2.2.1 Android Things

Es una plataforma, levantada en 2016, enfocada en el desarrollo de soluciones de tipo IoT que se presentan en un sistema operativo del mismo nombre. El sistema operativo en sí es una versión de Android con capacidades extras como lo son el control de sensores y hardware de diferente tipo (Google, n.d.-j). Debido a su naturaleza, como SO (Sistema Operativo) basado en Android, su desarrollo se lo realiza de la misma

manera como se realiza una app móvil, permitiendo alta facilidad y flexibilidad en las herramientas usadas (Google, n.d.-j).

Actualmente permite el desarrollo en Raspberry Pi 3 Model B y NXP Pico i.MX7D. El enfoque de la plataforma está en el prototipado y desarrollo y no el despliegue de soluciones a producción (Google, n.d.-j).

El enfoque de Android Things, en la actualidad, se encuentra limitado a pantallas inteligentes y altavoces. Aun así, se presenta como una herramienta altamente útil para el prototipado de soluciones de IoT en una manera sencilla (Miranda, n.d.).

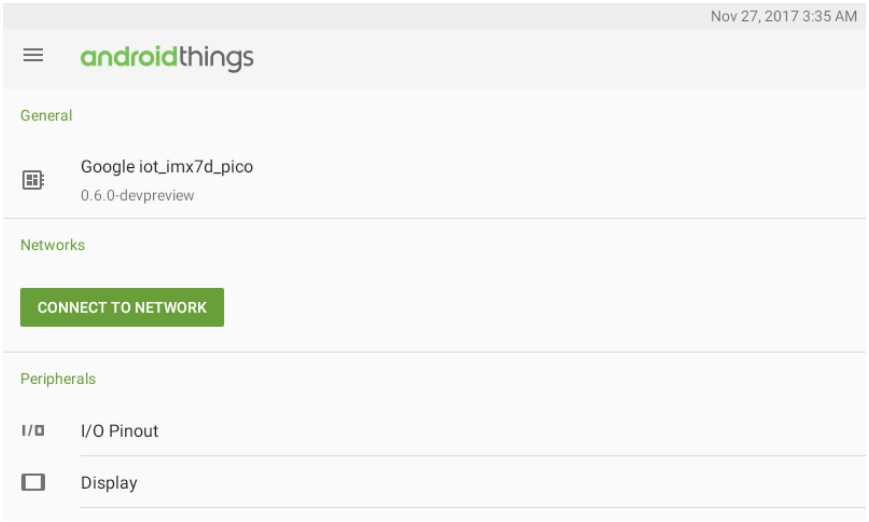

***Figura* 7.** UI principal de Android Things,

Fuente: (Google, n.d.-a)

3.2.2.2 Raspbian

El sistema operativo base para dispositivos como la Raspberry Pi. Es un sistema operativo, basado en Linux, derivado de Debian para computadores con arquitectura

ARM (RASPBERRY PI, n.d.-b). Presenta una amplia variedad de herramientas para la enseñanza y el desarrollo de aplicaciones informáticas de todo tipo, incluidas aquellas posibles con la Raspberry Pi.

Su principal enfoque es en el uso para la educación y prototipado de aplicaciones de todo tipo en Raspberry Pi (RASPBERRY PI, n.d.-b).

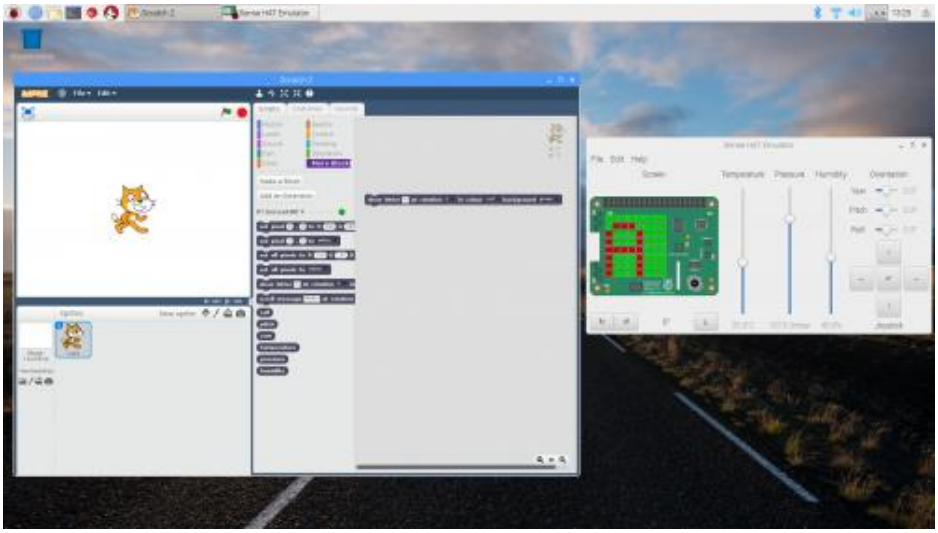

Figura 8. UI principal de Raspbian,

Fuente: (RASPBERRY PI, n.d.-d)

3.2.3 Lenguajes de Programación

3.2.3.1 Kotlin

Kotlin es un lenguaje de programación de tipado estático que soporta programación orientada a objetos y funcional. Corre sobre la máquina virtual de Java y por lo tanto es compatible con el uso de código java y sus librerías (Heiss, 2013). También puede ser compilado a código fuente de JavaScript (Heiss, 2013).

Es un lenguaje de programación ampliamente usado en la programación de apps para dispositivos Android, debido a su relativa facilidad de uso frente a Java (Google, n.d.-i).

3.2.3.2 Python

Es un lenguaje de programación interpretado, multiparadigma, dinámico y fuertemente tipado. Permite el desarrollo de código simple, adaptable y rápido. Es sumamente útil para el desarrollo de prototipos y aprendizaje, aunque también es ampliamente usado en la industria (Python, n.d.).

3.2.3.3 JavaScript

Es un lenguaje de programación ligero e interpretado, principalmente orientado a objetos con funciones de primera clase. JavaScript es un lenguaje que se lo usa en sitios web para su contenido dinámico, aunque presenta varios usos fuera del navegador como lo es Node.JS (Mozilla, n.d.).

Este lenguaje está descrito por el estándar ECMAScript, que establece que es lo que JavaScript puede o no puede hacer. Esto permite a todos los navegadores mantener un entendimiento claro del funcionamiento del lenguaje (Mozilla, n.d.).

3.2.4 Servicios de la nube

3.2.4.1 Firebase

Es una plataforma web enfocada en el desarrollo de aplicaciones web y móviles y la integración entre estas. Presenta una amplia serie de servicios que permiten el manejo de datos, sincronización y administración de servicios sin manejos de servidor (Google, n.d.-e). Aunque su enfoque está en el desarrollo web y móvil, presenta una amplia

variedad de aplicaciones donde es necesario comunicaciones en tiempo real y despliegues rápidos de servicios web (Google, n.d.-e).

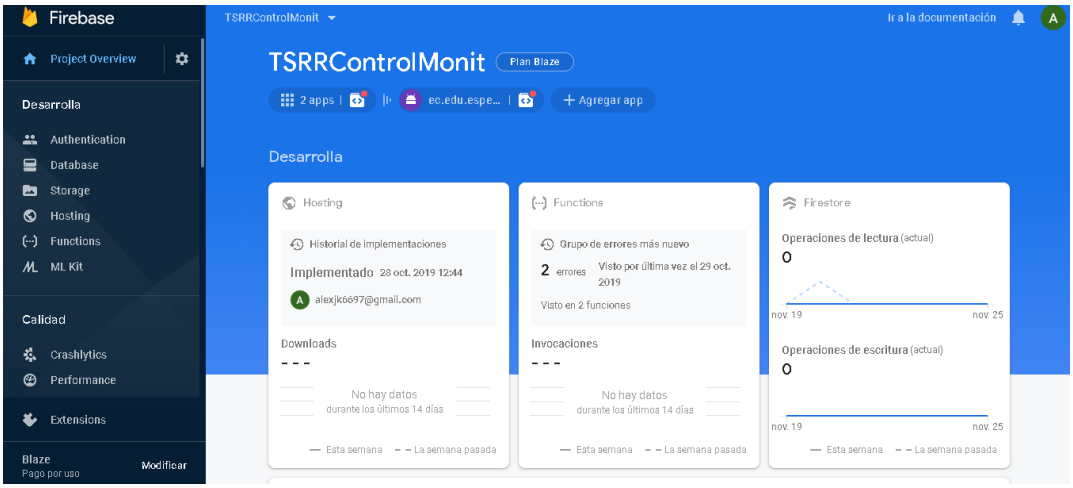

Figura 9. UI principal de la consola de Firebase,

Fuente:(“Firebase – Firebase console,” n.d.)

3.2.4.2 Real Time Database

La base de datos en tiempo de real de Firebase es una de las principales funcionalidades que tiene Firebase. Esta base de datos es de tipo NoSQL y guarda sus datos en formato JSON. Los datos almacenados tienen la capacidad de sincronizarse con los clientes en tiempo real, queriendo decir que cualquier acción sobre los datos se ve replicada (Google, n.d.-h).

Es particularmente mente para mantener información compartida en una amplia variedad de nodos, dispositivos o elementos entre sí, sin necesidad que estos tengan una comunicación directa (Google, n.d.-h). De esta forma un dispositivo puede afectar el comportamiento a base de la información que se establezca en la base de datos.

Esta base está construida con un enfoque en responsividad y velocidad de respuesta, por lo tanto las estructuras de datos almacenadas es relativamente sencillas y por lo tanto carecen de la capacidad de realizar consultas extremadamente complejas o especializadas (Google, n.d.-h). Su principal uso se encuentra en mantener información básica y de rápido acceso a diversos dispositivos (Google, n.d.-h).

Presenta soporte para lenguajes como Switft, Objective-C, Java, Kotlin, JavaScript, Go, Python y C++ principalmente por medio de un SDK y uso de WebSockets. También puede ser accedido por medio de REST APIs en cualquier lenguaje no soportado directamente (Google, n.d.-h).

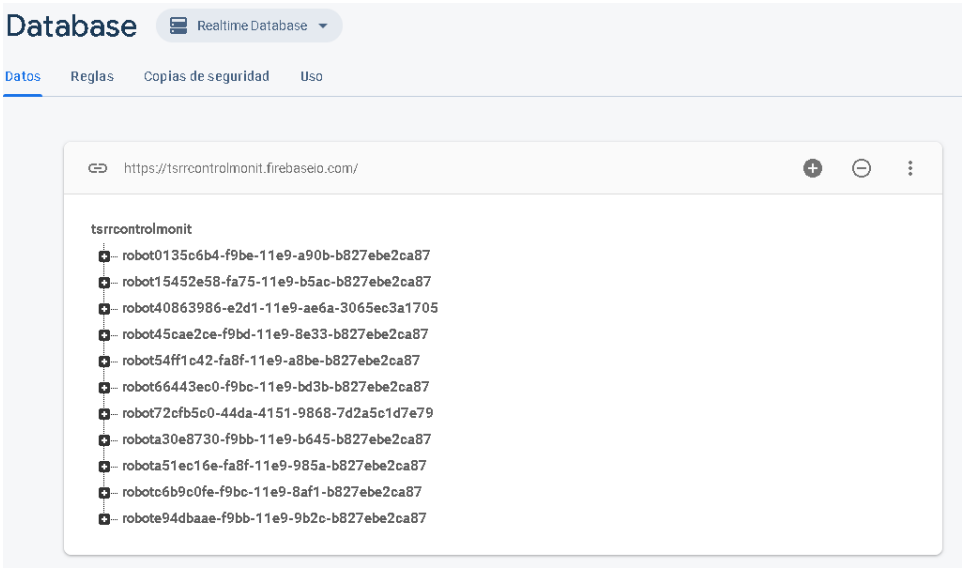

Figura 10. Estructura de la base de datos en tiempo real dentro de la consola de Firebase,

Fuente: (“Firebase – Firebase console,” n.d.)

3.2.4.3 Firestore

Firestore es una base de datos NoSQL de tipo clave-valor/documental que se administra sin ningún servidor o implementación a través de la plataforma de Firebase. Esta base se basa en la generación de colecciones y documentos a través de un modelo de datos dinámico y flexible (Google, n.d.-c). Permite el manejo de estructuras y consultas complejas. Presenta la capacidad de escalar a altas capacidad de manera sencilla (Google, n.d.-c).

Se emplea esta base de datos como un tipo de almacenamiento de base de datos equivalente a una base NoSQL tradicional, a diferencia de la sincronicidad y simplicidad de la base de datos en tiempo real (Google, n.d.-b).

Presenta un soporte con un SDK en lenguajes como Node.JS, Java, Kotlin, Python y Go; y REST y RPC APIs para otros lenguajes (Google, n.d.-c).

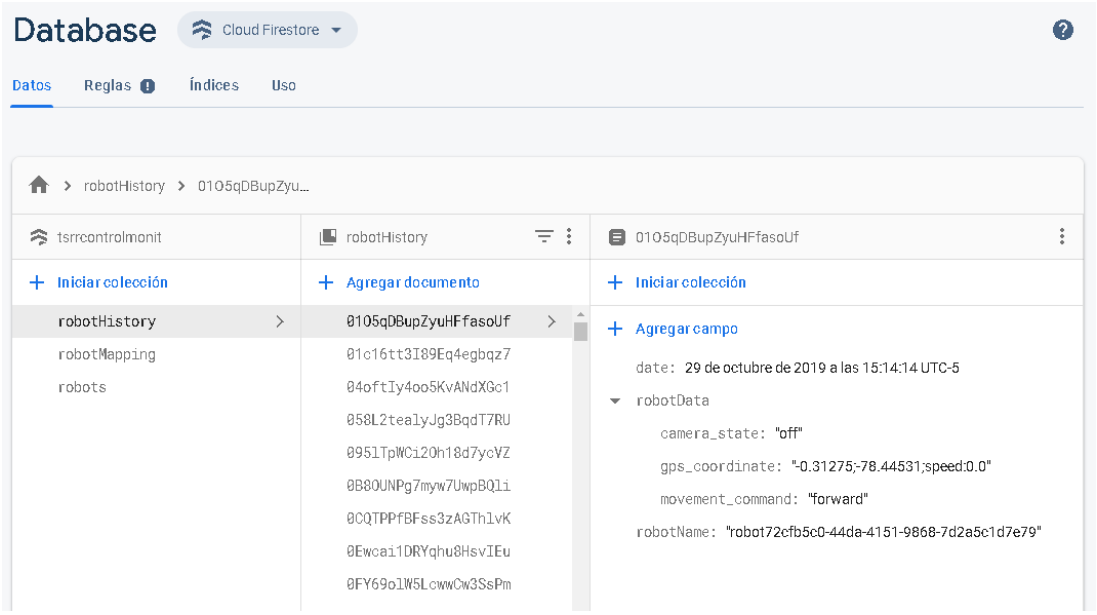

Figura 11. Estructura de la Firestore dentro de la consola de Firebase,

Fuente: (“Firebase – Firebase console,” n.d.)

3.2.4.4 Functions

Es un servicio de Firebase y del Google Cloud Platform que permite el correr código de backend, sin la necesidad de publicarlo en un servidor, a base de funciones específicas que se activan al momento de que existan diferentes eventos (Google, n.d.-d).

Los eventos a los que puede responder el usuario es una mezcla de eventos de servicios de la propia plataforma de Firebase y Google Cloud Plataform con eventos generados por el propio usuario (Google, n.d.-d).Estos eventos pueden ocurrir con acciones de:

- Firestore
- Real Time Database
- Remote Config
- Test Lab
- Authentication
- Google Analytics
- Crashlytics
- Cloud Storage
- Cloud Pub/Sub
- HTTP Triggers (GET, POST, PUT, DELETE, etc.)

Este tipo de funciones permiten la construcción de un BackEnd completo sin necesidad de levantar un servidor, la integración con otros servicios y la respuesta a acciones de otros servicios en el funcionamiento de un sistema (Google, n.d.-d). Claro

está, que su alcance de respuesta de eventos se limita a los servicios de Firebase y solicitudes HTTP (Google, n.d.-d).

Todas las ejecuciones de las funciones son registradas y se pueden revisar su estado, errores, logs y rendimiento (Google, n.d.-d).

Cloud Functions, se construyen con el uso de JavaScript con Node.JS (Google, n.d.-d).

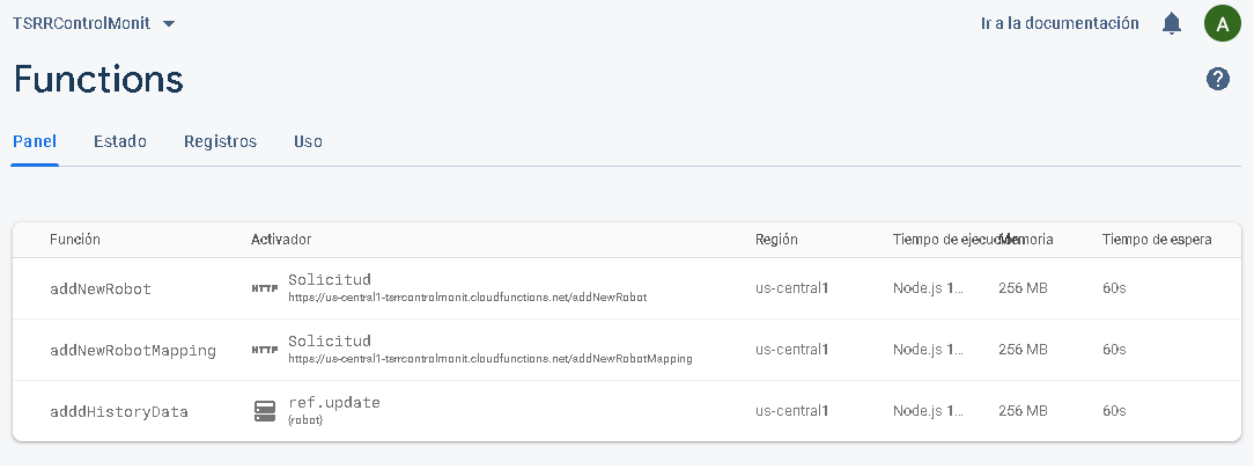

Figura 12. Estructura de la Cloud Functions dentro de la consola de Firebase,

Fuente: (“Firebase – Firebase console,” n.d.)

3.2.4.5 Hosting

Es un servicio de Firebase empleado para el hosting de web apps, contenidos dinámicos y estáticos, y microservicios sin una administración directa de servidores para su entrega (Google, n.d.-f).

Firebase Hosting se presenta como una plataforma administrativa para el despliegue de apps web y similares, en los cuales es posibles: publicar, editar, monitorizar

y configurar un sitio o aplicación web. Presenta una integración completa con otros servicios de Firebase (Google, n.d.-f).

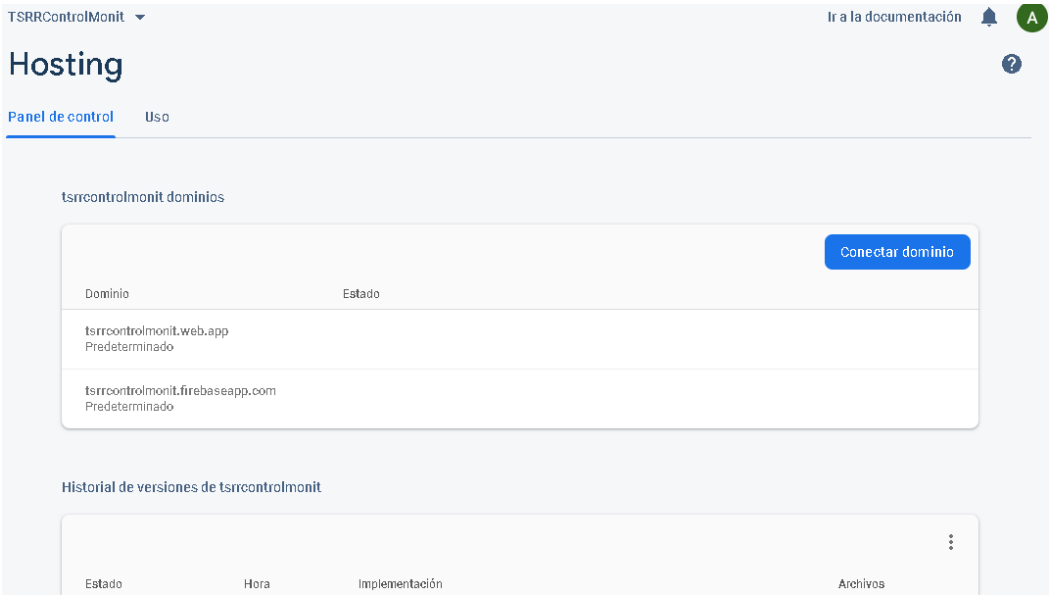

Figura 13. UI general de los servicios de hosting dentro de la consola de Firebase,

Fuente: (“Firebase – Firebase console,” n.d.)

3.3 DISEÑO Y ARQUITECTURA

3.3.1 Especificaciones de diseño

Para que la arquitectura funcione de una manera adecuada y efectiva, el diseño se encuentra limitado y definido por una serie de aspectos funcionales que son:

- Poder controlar y monitorear una cantidad ilimitada de robots, es decir que nuevos robots pueden entrar en el sistema e inmediatamente estar disponibles para que puedan ser manejados.

- El controlar un robot implica que se puede controlar sus sensores, movimiento y componentes cuando este se encuentra activo en el sistema. Todo esto en tiempo real.
- El monitorear un robot implica que se puede leer los datos que den sus sensores y otros componentes (e.j. cámaras) en tiempo real.
- Los robots a manejar pueden ser variados y estos poseer diferentes configuraciones. Aunque todos estos robots deben tener una configuración específica que sea compatible con el sistema, cada uno puede tener diferentes sensores y componentes que otros no. El sistema debe ser capaz de identificar qué elementos tiene el robot y el robot tiene que conocer qué componentes tiene.
- La configuración de un robot se divide en dos partes. La primera es una descripción de todos los atributos que pueden ser monitoreados y controlados. La otra es un mapa de control donde se especifica que botones o acciones del cliente afectan el control del robot.
- Al momento de controlar al robot se debe tener diferentes opciones disponibles, sea por medio de un dispositivo como un teclado o control, una interfaz gráfica o una aplicación móvil.
- El monitoreo del robot debe ser visualizado por medio de una interfaz que muestre de manera sencilla todos los datos capturados y en tiempo real del robot.
- Cuando existan más de un robot que tenga que controlado al mismo tiempo, debe de ser posible cambiar el control de un robot a otro en cualquier momento y a su vez debe ser posible controlarlos al mismo tiempo. Lo mismo ocurre con el monitoreo.

- Toda incursión que realiza un robot en específico debe ser almacenada en un historial para poder analizarla o acceder a esta posteriormente.

Cada una de estas funcionalidades están representadas de manera gráfica en la **figura 14.**

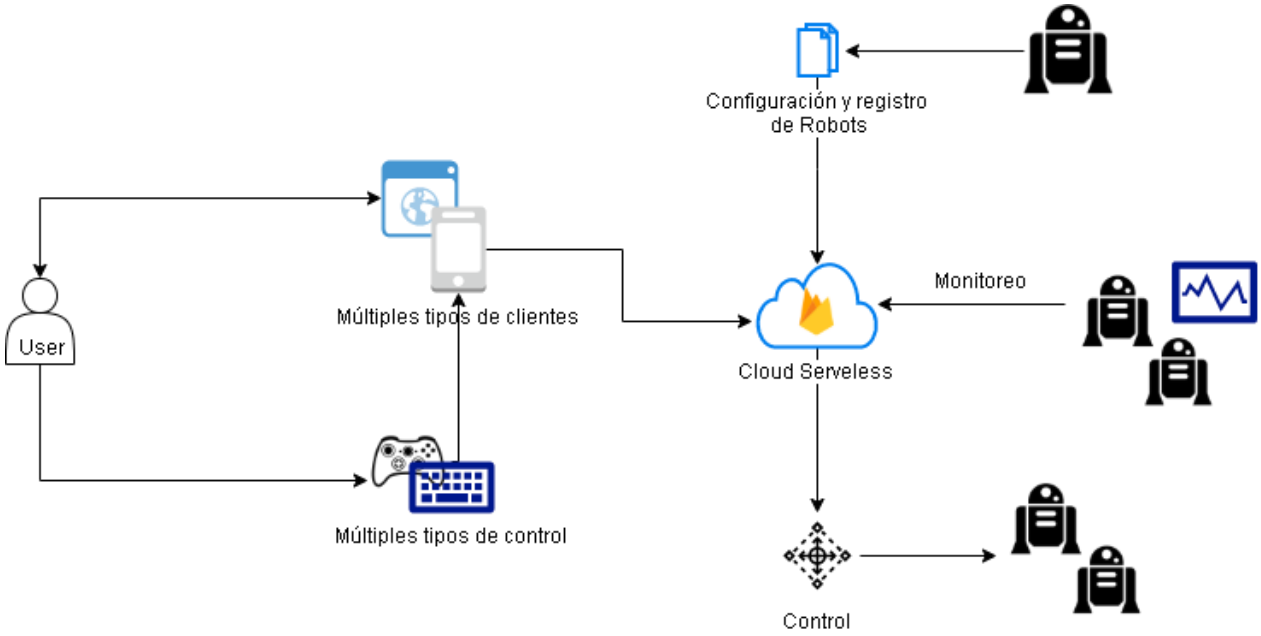

Figura 14. Funcionalidades permitidas por la arquitectura

Con las funcionalidades establecidas se determinó una arquitectura basada en un modelo de N capas (Buschmann, 2013; Marston, 2012). Donde en este caso existen 4 capas principales: Capa de presentación, Capa de almacenamiento o persistencia a datos**,** Capa de lógica de negocio, Capa de robots.

3.3.2 Capa de presentación

Esta capa tiene como elemento principal, la presentación de las interfaces y herramientas necesarias para que un usuario sea capaz de controlar diversos robots y a

su vez monitorear sus estados. Esta capa se presenta a través de una aplicación web que provee las siguientes capacidades:

- Inspeccionar todos los robots registrados en el sistema (**figura 15**).

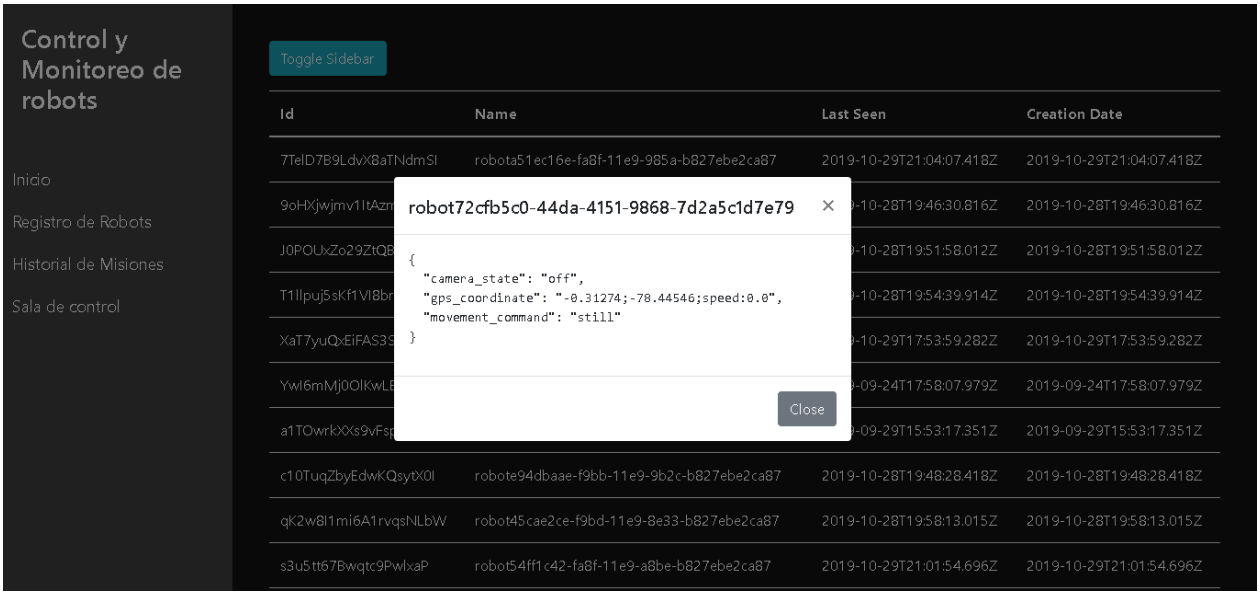

Figura 15. Robots registrados en el sistema

- Controlar y monitorear en tiempo real el robot (**figura 16**).

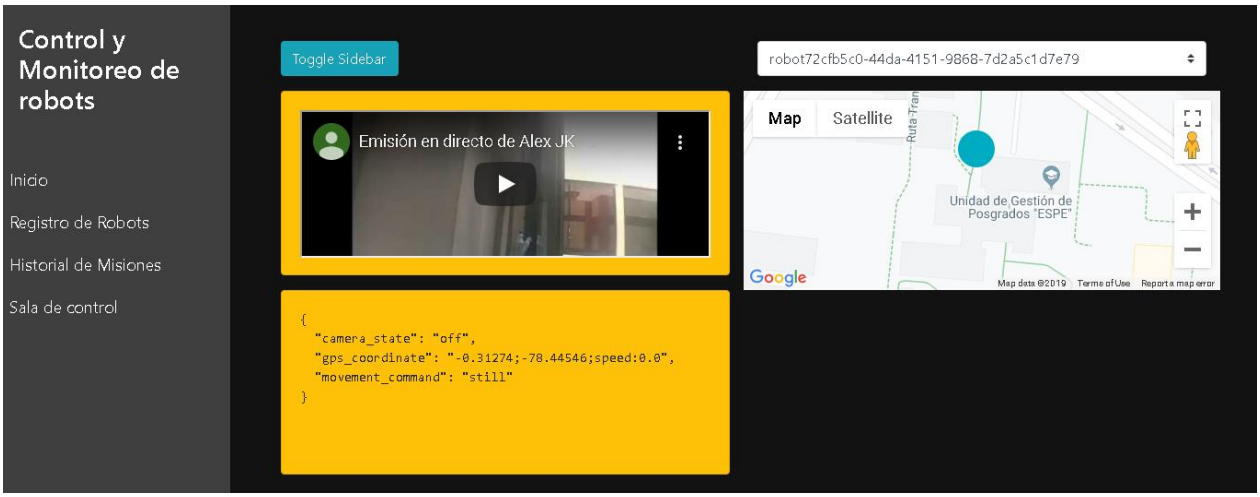

Figura 16*.* Representación básica del control de robots

- Acceder a una historia de acciones a lo largo del tiempo de robots en específico (**figura 17**).

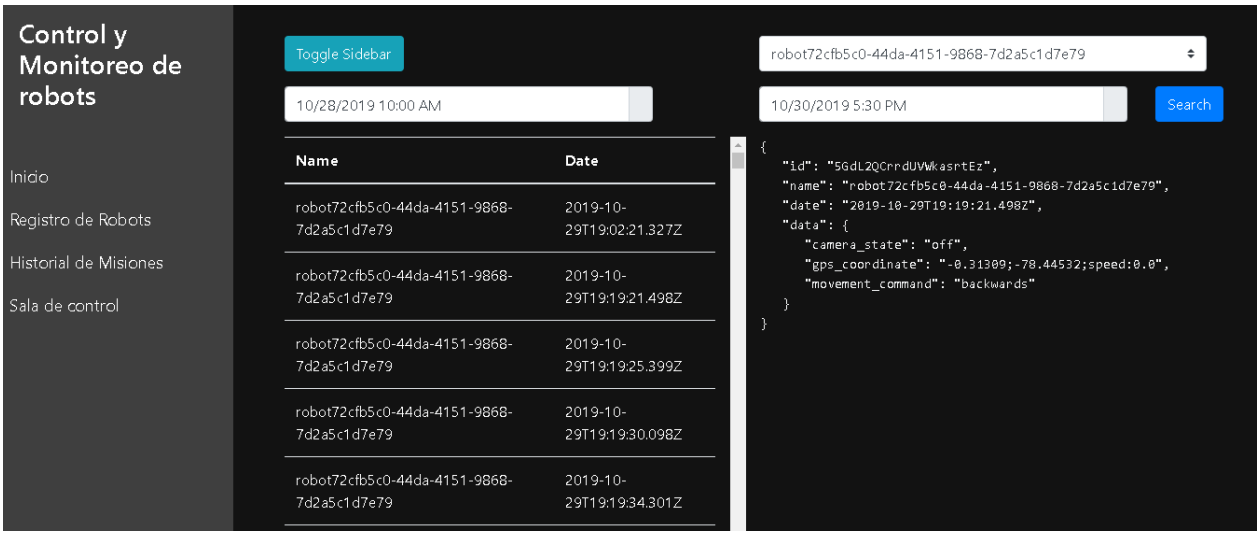

Figura 17. Historial de acciones de robots

Al ser una aplicación web, esta capa está principalmente realizada con el uso de HTML, CSS y JavaScript. La estructura de esta capa se muestra en la **figura 18**.

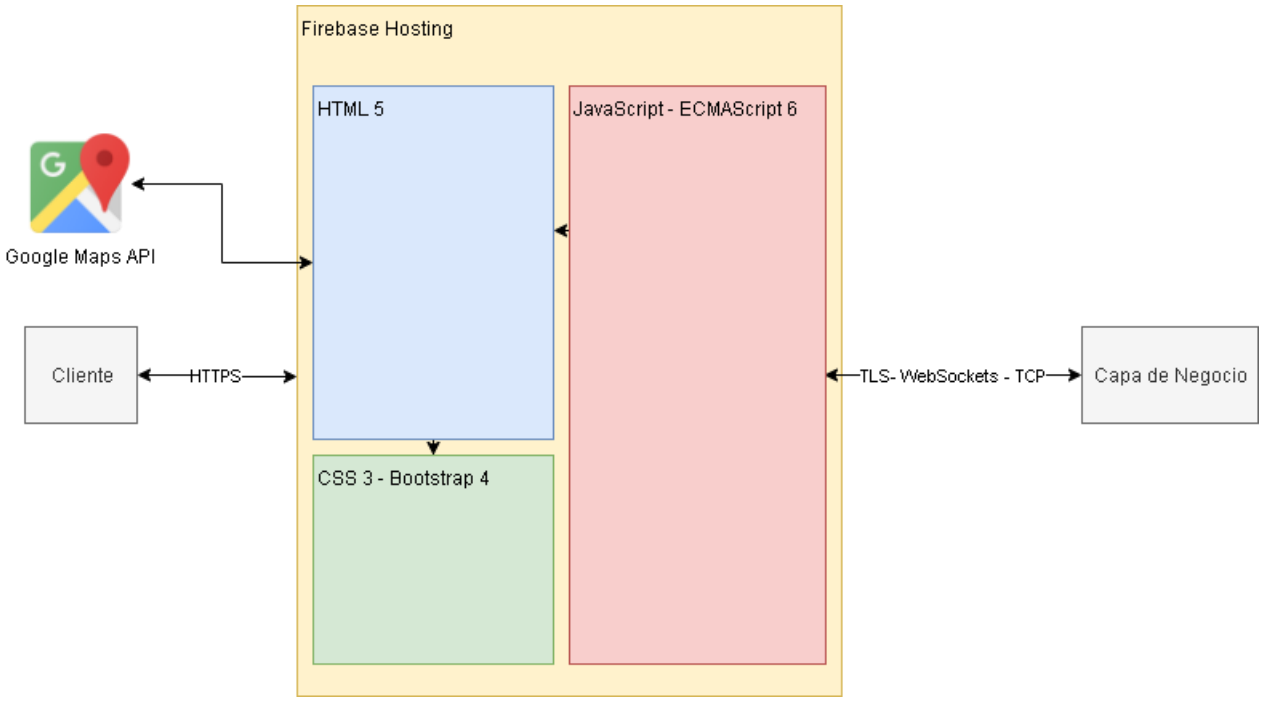

Figura 18. Estructura de la capa de presentación

3.3.3 Capa de almacenamiento o persistencia de datos

Esta capa hace referencia específicamente a las diferentes bases de datos que se usan para almacenar la información que va a ser utilizada por el sistema de control y monitoreo; sea en tiempo real o no. Esta capa se hizo uso de Firebase Realtime Database y Firestore.

La base de datos en tiempo real se usó para la entrega en tiempo real de información capturada por diversos robots y para el control de robots.

Firestore, en cambio, se encargó de almacenar a robots registrados, historial de acciones y mapas de control para uso de mandos en el control de robots.

La **figura 19** muestra de manera general el esquema de esta capa.

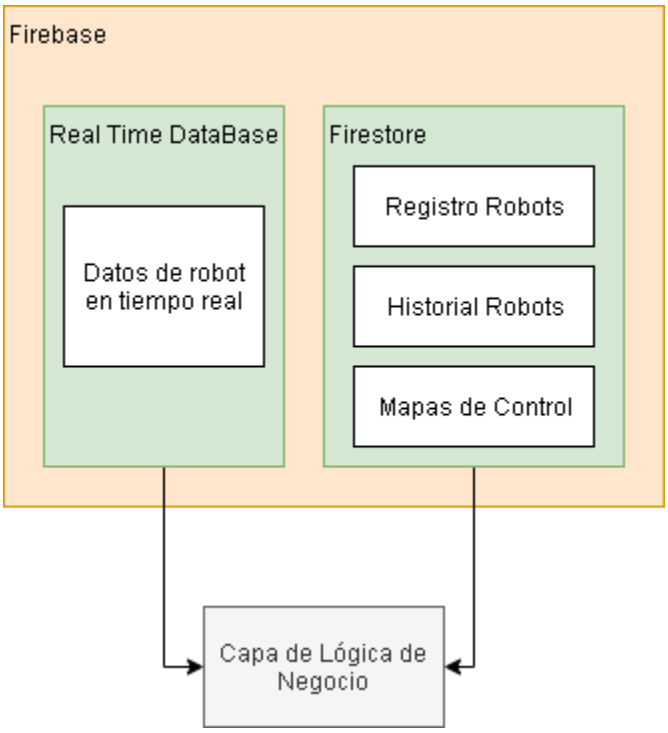

Figura 19. Estructura de la capa de datos

3.3.4 Capa de lógica de negocio

Esta capa presenta los medios por los cuales la capa de presentación y de robots es capaz de comunicarse con la base de datos en tiempo real y Firestore. Esta capa presenta diversas funciones serverless para responder a diferentes eventos y asegura la comunicación de los robots y la capa de presentación con las bases de datos.

Esta capa permite que se registren acciones de los robots, registrar y configurar robots, y asegura que se puedan escribir y leer datos en tiempo real de cada robot.

La **figura 20** presenta la estructura general de esta capa.

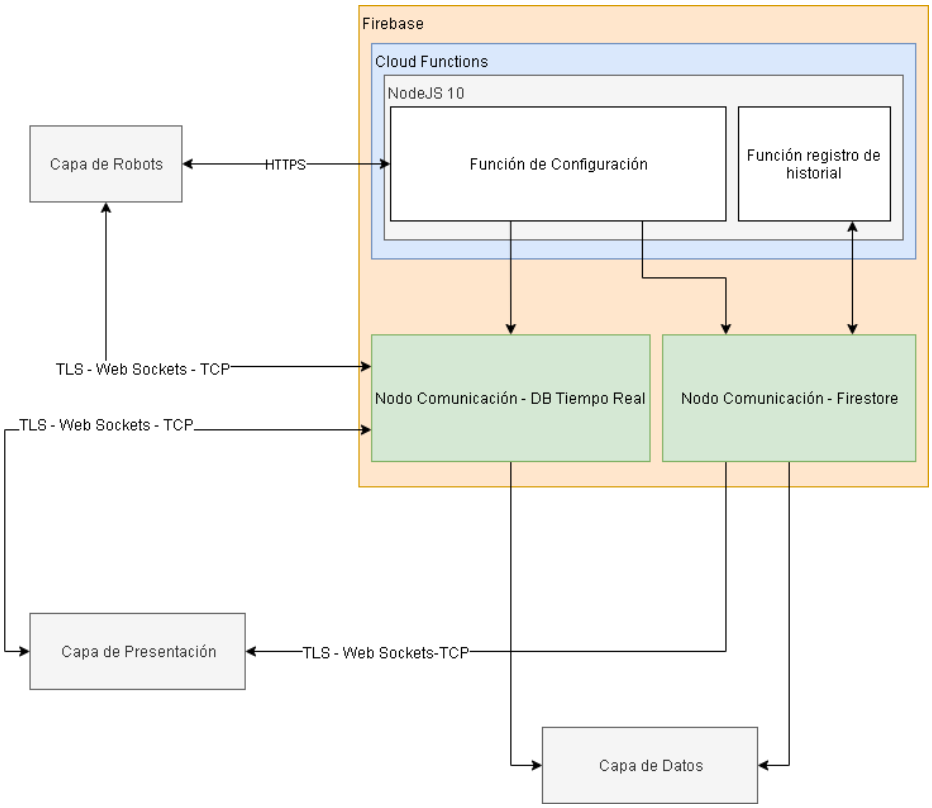

Figura 20. Estructura de la capa de lógica de negocio

3.3.5 Capa de robots

Esta capa es la que representa a los robots y su comunicación con el resto de la arquitectura. Se compone de diferentes elementos de hardware y software que permiten el control de robots y que este envíe información capturada del entorno.

Para que esta capa funcione adecuadamente, el robot debe estar construido de una manera específica. En primer lugar, el robot debe tener la capacidad de poder comunicarse con los servicios de Firebase a través de internet y por medio de un SDK. Segundo, el robot debe presentar al momento de su configuración 2 elementos: una descripción de funcionalidad y un mapa de control. Estos elementos serán usados por

las capas anteriores para dirigir y obtener los datos del robot. Tercero, el robot debe enviar o recibir información a base de su descripción de funcionalidad.

La descripción de funcionalidad tiene el siguiente formato:

```
robotid @@@
{
      “control_parameter1”: value,
      “control_parameter2”: value,
      “monitor_parameter1”: value,
      …
}
```

El mapa de control tiene el siguiente formato:

```
robotid@@@
{
      {
            “buttons”: {
                  "X”: {
                        "collection":"control_parameter2",
                        "value":value
                   },
                  ...
        },
        "directions”: {
                  "up”: {
                         "collection":"control_parameter1",
                         "value":value
                   },
                  …
      }
}
```

La **figura 21** presenta la estructura general de esta capa.

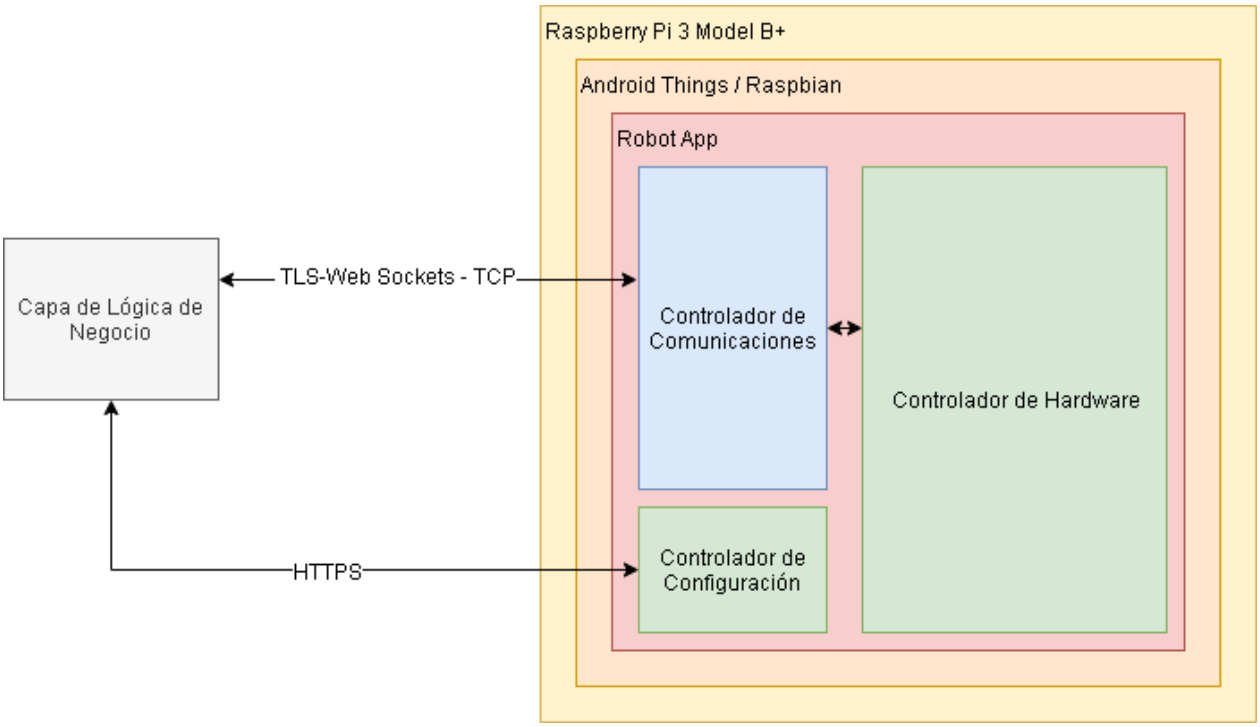

Figura 21. Estructura de la capa de robots

3.3.2 Video Streaming

La capacidad de realizar video streaming no es un elemento que esta explícitamente expuesto a la arquitectura y tampoco es una parte de esta. Aun así, es un elemento importante en el control y monitoreo de un robot para poder saber qué es lo que ve. Por esta razón, los robots pueden conectarse libremente a cualquier servidor RTMP para poder transmitir video a través de una cámara.

3.4 ESQUEMA DE ARQUITECTURA

La arquitectura se define a base de las estructuras y/o esquemas de cada una de las capas, y sus interacciones entre sí. De esta forma la arquitectura surge como una

composición de la **figura 18**, **figura 19**, **figura 20**, **figura 21**. De esta forma la arquitectura está representada por la **figura 22.**

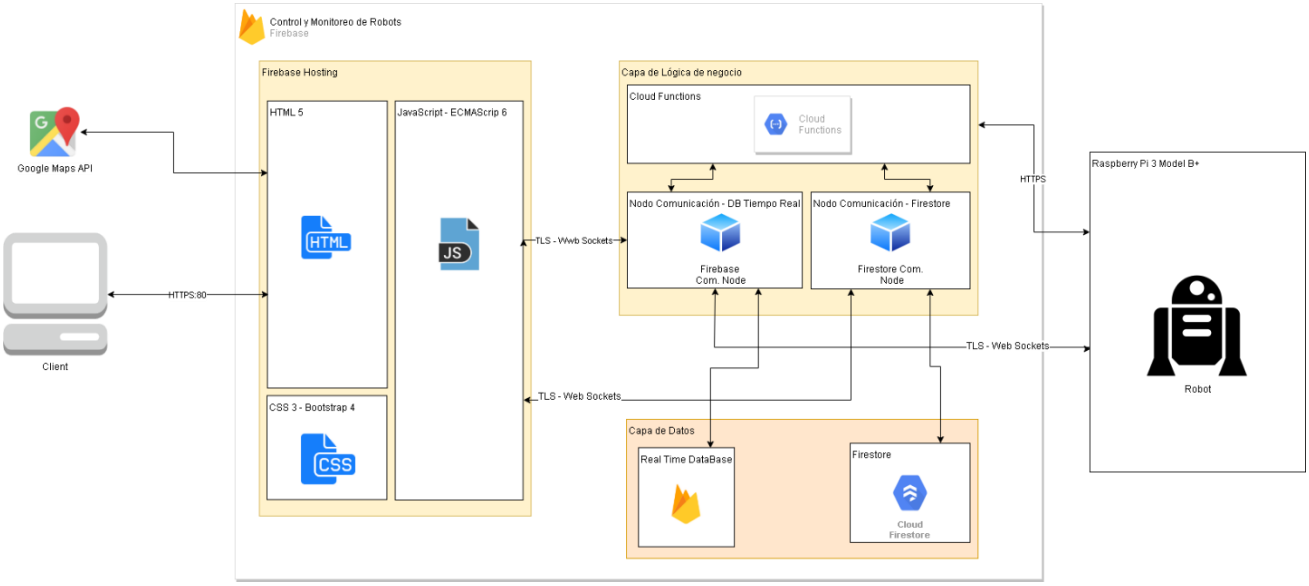

Figura 22. Arquitectura de todo el sistema del control y monitoreo de robots

3.5 SEGURIDAD

La seguridad de la arquitectura se garantiza gracias al uso de Firebase, que presenta diversos elementos de seguridad que son configurables y adaptables en la transmisión y comunicación de información.

En primer lugar, todas las conexiones que las realizan con el uso de TLS, es decir que todas las conexiones son seguras y se encuentran encriptadas y por lo tanto no es posible interceptar datos fácilmente.

Las funciones que presentan endpoints tienen la capacidad de agregar diversos medios de autenticación, tales como JSON Web Tokens o credenciales propias de

Firebase. Con esto, solo quienes tengan estos elementos podrías hacer uso de los enpoints.

Las bases de datos se aseguran través de una serie de reglas que especifican quienes pueden leer y escribir sobre las bases de datos. Se puede especificar de manera detallada el medio para acceder a la base y quienes pueden acceder a lugar específicos de la base de datos. Es decir que se puede establecer un acceso granulado por cada robot o un acceso general a base de las credenciales de Firebase.

El acceso a los servicios de Firebase se los hace a través de una cuenta de Google y a través de credenciales y llaves secretas. Sin estas no es posible cambiar el funcionamiento de los servicios o sus configuraciones.

Por último, los datos de configuración que inicializan, aunque fácilmente obtenibles de los robots, no tienen mayor uso si no se tiene acceso a los servicios, endpoint o bases de datos de la arquitectura; ya que estos elementos de configuración se gestionan desde allí.

CAPÍTULO IV

VALIDACIÓN DE ARQUITECTURA

4.1 INTRODUCCIÓN

Una vez que se tenga una representación a través de un prototipo de la arquitectura de monitoreo y control planteado, es necesario probarlo y validarlo para determinar su utilidad en el uso de operaciones de búsqueda y rescate urbano.

La validación del prototipo se debe realizar en un ambiente controlado y con un enfoque principal en QoS.

El uso de un ambiente controlado, es decir de laboratorio, es debido a que la arquitectura propuesta que se propone no tiene indicios de pruebas pasadas y por lo tanto algún indicio de su utilidad. Un ambiente controlado permitió establecer fácilmente en qué aspectos la arquitectura propuesta funciona correctamente y en qué aspectos existieron limitaciones o problemas.

El enfoque de realizar pruebas de QoS; latencia, rendimiento, transferencia de datos; se debe a que estas permitieron establecer la utilidad de la arquitectura en diferentes condiciones y los requisitos de esta arquitectura para funcionar. Con esto y comparando con el proceso que requiere una operación de búsqueda y rescate, y con las necesidades que debe tener un sistema de control para un uso adecuado; se pudo conocer la utilidad de la arquitectura en el control y monitoreo de robots en operaciones de búsqueda y rescate.

4.2 HERRAMIENTAS

4.2.1 Herramientas de monitoreo y análisis

Las herramientas monitoreo son necesarias para poder capturar los diferentes datos enviados, los niveles de rendimiento y los tiempos de respuesta al momento que se realizan las pruebas. Las diferentes herramientas se emplean para analizar los entornos del cliente web, los robots y la nube en sí. Las herramientas para usar son:

4.2.1.1 Wireshark

Es un software para el análisis y captura de tráfico de red (Wireshark, n.d.). En este caso, se lo emplea para la captura de datos desde el cliente web. También este software es esencial para el análisis de toda la información capturada desde los robots y el cliente.

4.2.1.2 TCP Dump

Es un software que se ejecuta en línea de comandos que tiene por objetivo el análisis de paquetes enviados y recibidos de red (“TCPDUMP/LIBPCAP public repository,” n.d.; tcpdump, n.d.). En este caso, esta herramienta resulta esencial para el análisis de tráfico de red desde el punto de vista de los robots usados.

4.2.1.3 Firebase Analytics

En este caso firebase analytics se refieren a las diferentes estadísticas de uso de cada uno de los servicios usados (“Firebase – Firebase console,” n.d.). Firebase provee estadísticas sencillas del uso de sus diferentes servicios que nos permite establecer de manera general el rendimiento de nuestros servicios y aplicaciones (“Firebase – Firebase

console," n.d.). Este elemento se usará para analizar el rendimiento de los servicios en la nube.

4.2.2 Herramientas de red

Estas herramientas se refieren a todos los elementos necesarios en aspectos de hardware de red para poder establecer la topología de pruebas. En este caso se requiere el uso de:

- 1 Router - En este caso se ha hecho uso del router IPTECOM CDW531AM-002
- 2 Router/Access Points - Se ha hecho uso del router NEXXT Modelo ARN04904U1 y el router CISCO Linksys Modelo WRT310N V2. Algo a tomar en cuenta es que uno de los access points debe tener la capacidad de regular de manera detallada el ancho de banda a la que tiene acceso los dispositivos que se conecten a este. Esto es necesario para así poder generar diferentes pruebas a base de diferentes condiciones de ancho de banda de manera precisa.

4.3 ENTORNO DE PRUEBA

4.3.1 Entorno general

El entorno de prueba presenta una variedad de elementos que se deben considerar en cuenta en base a la arquitectura, específicamente la capas que presenta la arquitectura y las funcionalidades que se tienen. De esta forma se considera lo siguiente:

- 3 capas principales en la arquitectura: El cliente, los servicios serverless y el robot.
- 3 funcionalidades principales: Registro de robots, Control de robots, Monitoreo de robots.

De esta forma, el entorno de pruebas puede ser descrito para todos los casos de prueba con la presencia de un entorno físico de pruebas y una serie de procesos que deben existir.

4.3.2 Entorno físico

El entorno consta de 2 redes diferentes que no tengan una conexión entre sí, es decir que deben estar totalmente independiente. Ambas redes deben contar con una conexión a internet y presentar de una access point con conexión WiFi. Cada uno de los nodos conectados en la red deben tener la capacidad de monitorear el uso de red. Además de eso, se debe establecer un monitoreo de uso en los servicios serverless usados. De esta forma tenemos:

- Red 1 - Red Cliente: Donde el cliente controlador se encuentra
- Red 2 - Red Robots: Donde uno o más robots se encuentran en funcionamiento dependiendo del caso de prueba.
- Entorno de Nube - Serverless: Donde están los servicios usados para el control y monitoreo alojados.

De esta forma se tiene el esquema general representado en la **figura 23**.

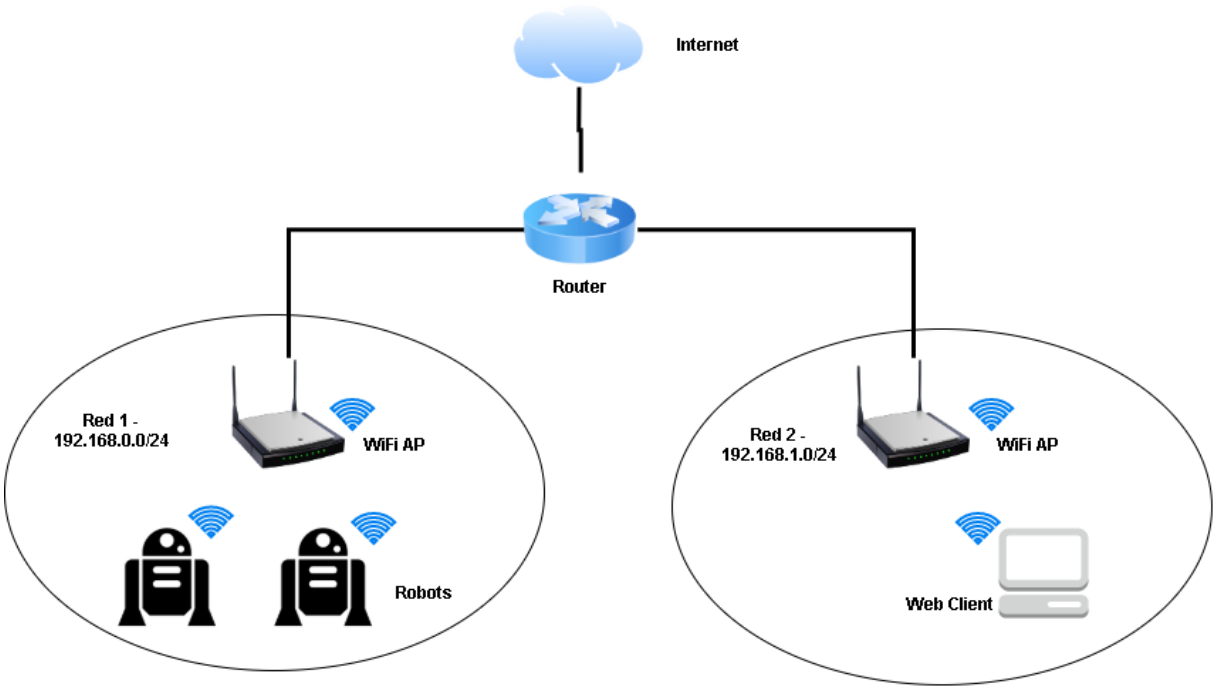

Figura 23 Esquema general del entorno físico de pruebas

4.3.3 Procesos

Sin importar el caso de prueba se deberá seguir un estándar en que elementos generales serán probados, más no en el contexto que serán probados. En general, el proceso se enfoca en los robots en sí, y el proceso de prueba deberá seguir el siguiente esquema:

- Pruebas de configuración inicial del robot: Se deberá empezar con un robot no registrado en la red y ejecutar su configuración inicial y su registro en el sistema de control y monitoreo.
- Pruebas de monitoreo y control: Una vez inicializado el robot se procederá a que empiece el monitoreo y control. En este caso el control se basará en el movimiento

y el uso de la cámara y el monitoreo sobre los datos que envíe el robot para ser almacenados en el sistema.

4.4 CASOS DE PRUEBA

4.4.1 Generalidades

Para poder tener una visión clara de la arquitectura en diferentes condiciones es necesario establecer diferentes casos de pruebas que validen la arquitectura en diferentes condiciones. Las pruebas, que se basan en medir diferentes niveles de QoS (Quality of Service, en español Calidad de Servicio) de la arquitectura se centran en la manipulación de las condiciones de red y comunicación que son esenciales para el funcionamiento de la arquitectura. De esta forma, se estableció dos casos de prueba fundamentales:

- Pruebas basadas en la variación del ancho de banda en el acceso a internet.
- Pruebas basadas en la variación de la intensidad de señal de la red WiFi.

4.4.2 Prueba de variación de velocidad de internet

En este caso se propone realizar los procesos ya descritos en diferentes niveles de ancho de banda o velocidad en la red de robots. De esta forma se pretende determinar las diferencias que pueden surgir en la comunicación de los robots con la nube y su control con el cliente web. Además, se pretende determinar cómo es afectado el rendimiento en el streaming de video.

La forma de realizar esta prueba es ejecutar sus procesos en 5 velocidades diferentes: 100KBps, 50KBps, 25KBps, 10KBps y 5KBps.

Con el uso de TCPDump y Wireshark, capturar todos los paquetes referentes a la comunicación en la nube que se transmitan a través de las redes. De allí, evaluar los resultados de las capturas y poder llegar a conclusiones.

4.4.3 Prueba de variación de señal basado en intensidad de señal

En este caso se propone realizar los procesos ya descritos en diferentes niveles de intensidad de señal de WiFi. De esta forma se pretende determinar las diferencias que pueden surgir en la comunicación de los robots con la nube y su control con el cliente web. El cambio de la intensidad de señal se lo realiza se lo realiza en la red de robots.
La forma de realizar esta prueba es ejecutar sus procesos en 3 rangos de intensidad de señal diferentes: -30dbm a -50 dbm (Señal de alta calidad), -50 dbm a -70 dbm (Calidad Media), -70dbm al -90dbm (Calidad baja). Estos datos se encuentran determinados a base de las recomendaciones de (Moyers, 2015).

Con el uso de TCPDump y Wireshark, capturar todos los paquetes referentes a la comunicación en la nube que se transmitan a través de las redes. De allí, evaluar los resultados de las capturas y poder llegar a conclusiones.

4.5 RESULTADOS

4.5.1 Elementos a analizar

4.5.1.2 RTT

Se refiere a Round Trip Time y se refiere al tiempo de ida y vuelta de enviar un paquete y recibir su respuesta desde su receptor (“Definition: round-trip delay time,” n.d.). En otras palabras, es el tiempo desde que se inicia el envío de un paquete hasta que se recibe un ACK de confirmación de recepción de este. Es un data particularmente útil para determinar el rendimiento de una red y los tiempos de respuesta entre cliente y servidor.

4.5.1.3 Tiempos por petición HTTP

Se refiere al tiempo completo que una petición HTTP tarda en ser completada. A diferencia del RTT que mide el tiempo de entrega de un paquete. Este tiempo se refiere al tiempo de entrega de un mensaje completo.

4.5.1.4 Tamaño promedio por paquete

Se refiere al tamaño que tiene un grupo de paquetes de una misma conexión en promedio. Este dato nos permite conocer cómo se están enviando paquetes en una conexión, si su tamaño causa retraso y el ancho de banda mínimo para que puedan ser enviados.

4.5.1.5 Porcentaje de carga de servicios de bases de datos en tiempo real

Se refiere al porcentaje de procesamiento que se encuentra usando la base de datos durante intervalos de un minuto. Al llegar al 100% empiezan a existir problemas de rendimiento y velocidad. Este dato ayuda a determinar la cantidad máxima de peticiones o dispositivos que pueden estar conectados a la base de datos al mismo tiempo (Google, n.d.-h).

4.5.1.6 Transferencia de datos

Se refiere a la cantidad de datos que han sido descargados de las bases de datos, se en KiloBytes o en número de lecturas. Ayuda a determinar el uso de las bases de datos por elemento conectado y predecir costos en el uso (Google, n.d.-h).

4.5.1.7 Almacenamiento de datos

Se refiere a la cantidad almacenadas o datos guardados en las diferentes bases de datos usadas, sea en KiloBytes o número de escrituras. Permite establecer una proyección de uso de los servicios por elemento conectado (Google, n.d.-h, n.d.-c).

4.5.2 Resultados de pruebas basadas en velocidad

4.5.2.1 Configuración e inicialización de robots

La configuración e inicialización de robots se refiere a la configuración o conexión por primera vez de un robot y su introducción dentro de la arquitectura. Este proceso ocurre a base de una serie de peticiones HTTP/TLS para enviar datos desde el robot a los servicios en la nube y almacenar sus datos. Se registraron estas peticiones en diferentes condiciones de ancho de banda, resultando en la **tabla 4**. A representa al robot y B el endpoint conectado a la nube.

Tabla 4

Resultados de configuración e inicialización de robots

VELOCIDAD	100 KBps	50KBps	25KBps	10KBps	5KBps
AVG RTT A->B (ms)	0.49395	0.4940	0.5079	0.48365	0.6783
AVG RTT B->A (ms)	93.2906	86.9959	89.5462	89.53130	114.564
AVG Packet Size A->B (bytes)	141.63	141.63	141.63	141.63	141.63
AVG Packet Size B->A (bytes)	815.93	815.93	815.93	815.93	815.93
AVG Message Time (sec)	0.72045	0.6803	0.7580	1.11905	2.76535

A base de esto se determinó los gráficos para resultados de configuración e inicialización de robots tanto en RTT como en duración del mensaje, estos gráficos se encuentran en la **figura 24** y la **figura 25.**

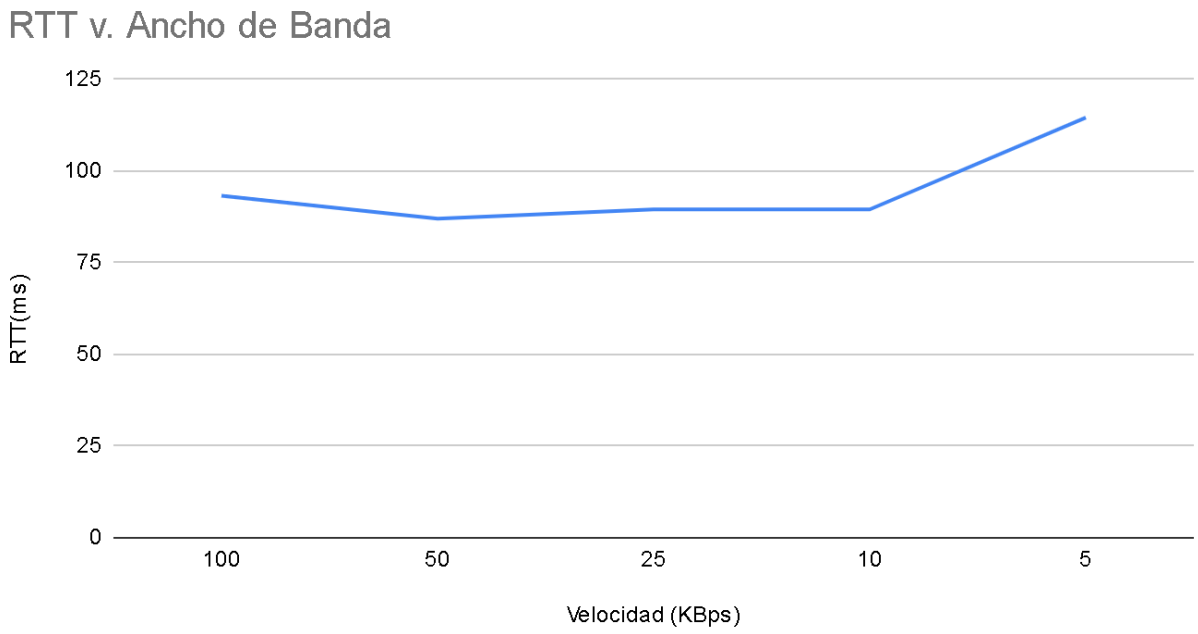

Figura 24. Resultados de configuración e inicialización de robots (RTT)

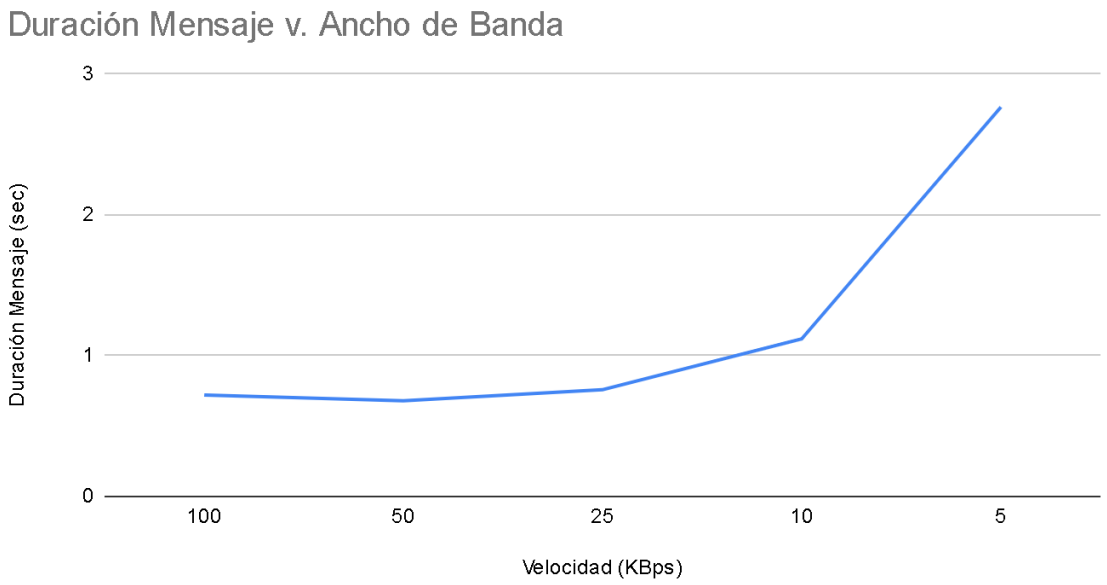

Figura 25. Resultados de configuración e inicialización de robots (Duración del mensaje)

Discusión

Con los resultados presentes podemos notar que el rendimiento de la red no presenta mayores cambios, a excepción de cuando el ancho de banda decrece a 5KBps y aun así este crecimiento es relativamente bajo. Por el otro lado la duración del mensaje crece a medida que el ancho de banda se reduce.

4.5.2.2 Control y monitoreo de robots

El control hace referencia al proceso de que el cliente web por medio de algún input comande o entrega acciones a realizar al robot, en este caso es el movimiento, en este caso coordenadas GPS. El monitoreo en cambio es la entrega de información por parte del robot hacia el cliente web. Todas estas acciones ocurren por medio del entorno en la nube. Durante estas pruebas se realizaron aquellas que solo media el control, y

otras que media el control y el monitoreo tanto con varios robots (2 en total) y con uno solo. Los datos fueron capturados tanto desde el punto del cliente como desde los robots. *A* representa al robot o al cliente y *B* representa al endpoint conectado a la nube.

En el caso de pruebas que solo presentan el control se obtuvieron los resultados presentes en la **tabla 5** y la **tabla 6**.

Tabla 5

Resultados de control de robots (Robot)

VELOCIDAD (KBps)	**100**	**50**	**25**	**10**	**5**
AVG RTT A->B (ms)	0.857859	0.7197	0.7081	1.10919	0.534134
AVG RTT B->A (ms)	89.862800	104.9890	90.1500	90.51580	91.807400
AVG Packet Size A->B (bytes)	78.87	77.26	76.73	84.39	74.75
AVG Packet Size B->A (bytes)	185.71	162.50	178.57	204.08	174.76

Tabla 6

Resultados de control de robots (Cliente Web)

VELOCIDAD (KBps)	**100**	**50**	**25**	**10**	**5**
AVG RTT A->B (ms)	3.32263	2.4723	2.9531	2.23348	2.601680
AVG RTT B->A (ms)	87.830700	87.4599	87.5805	87.03980	88.145800
AVG Packet Size A->B (bytes)	183.38	181.13	176.96	177.23	178.60
AVG Packet Size B->A (bytes)	94.65	96.86	95.93	95.69	97.05

A base de esto se determinó gráficos de análisis de RTT con el ancho de banda tanto en el robot como en el cliente; estos gráficos se visualizan en la **figura 26** y la **figura 27**.

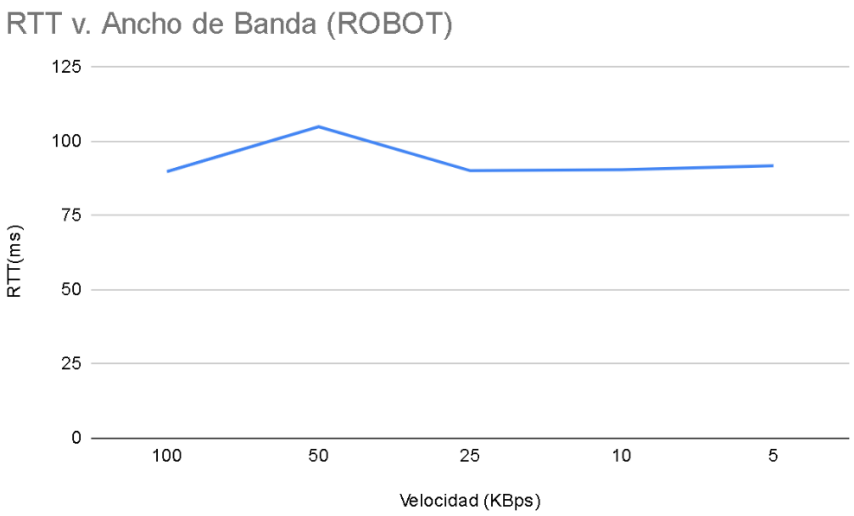

Figura 26. Resultados de control de robots (Robot)

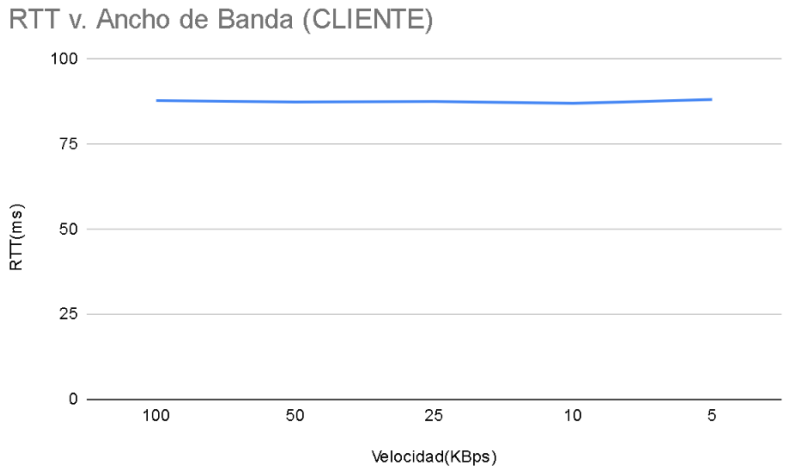

Figura 27. Resultados de control de robots (Cliente Web)

Discusión

El rendimiento de la comunicación tomando en cuenta RTT al momento de solo mantener control, es estable sin importar el ancho de banda, al menos hasta llegar a los

5KBps, teniendo ciertos aumentos en algunos casos. De esta manera, el rendimiento de los robots está en el rango de los 88 ms y los 110 ms. Claro está que el análisis se lo hace en la respuesta del servidor ante mensajes del robot, y es una aproximación a los RTT que esperaría el servidor cuando recibe respuesta del robot. Aunque los RTT se mantengan estables, los mensajes o comandos que reciba el robot mantendrían el mismo RTT mientras tengan el mismo tamaño del paquete enviado.

El cliente web mantienen un RTT estable en un rango de 86 ms y 89 ms sin importar el ancho de banda del robot. Esto es de esperar ya que la red del cliente web no recibió cambios de ancho de banda.

Tomando los dos datos de RTT del cliente web y el robot, se puede esperar que aproximadamente cada comando tendrá un retraso entre 174 ms y 199 ms, siempre y cuando no excedan el tamaño promedio de un paquete.

En el caso de pruebas que probaron el control y el monitoreo, los resultados se encuentran en la **tabla 7** y la **tabla 8**.

Tabla 7

Resultados de control y monitoreo de robots (Robot)

VELOCIDAD (KBps)	**100**	**50**	**25**	**10**	**5**
AVG RTT A->B (ms)	0.494481	0.9063	0.420036	0.48365	0.512101
AVG RTT B->A (ms)	95.945700	95.135	95.135	98.84170	125.334
AVG Packet Size A->B (bytes)	116.13	107.69	103.26	107.14	108.97
AVG Packet Size B->A (bytes)	276.19	192.31	190.22	196.43	192.31

Tabla 8

Resultados de control y monitoreo de robots (Cliente Web)

VELOCIDAD (KBps)	100	50	25	10	5
AVG RTT A->B (ms)	1.97926	2.3514	1.98785	2.75766	9.13448
AVG RTT B->A (ms)	86.531300	86.8249	86.6375	87.11700	204.631
AVG Packet Size A->B (bytes)	181.73	179.10	180.56	177.57	180.94
AVG Packet Size B->A (bytes)	96.29	96.53	96.46	96.28	101.91

A base de esto se determinó gráficos de análisis de RTT con el ancho de banda tanto en el robot como en el cliente, que se visualiza en la **figura 28** y la **figura 29**.

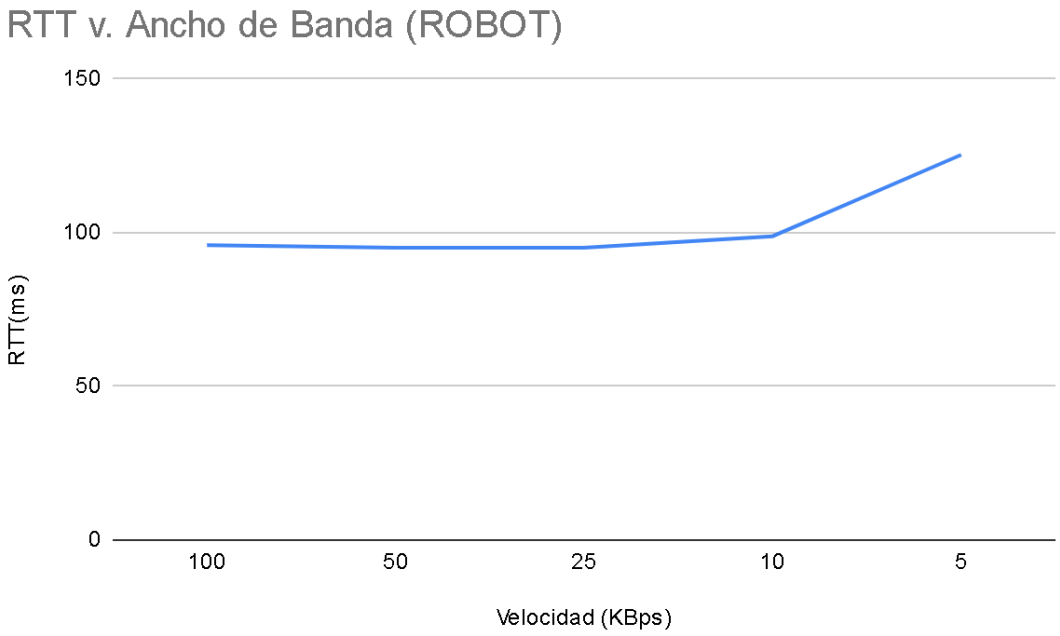

Figura 28. Resultados de control y monitoreo de robots (Robot)

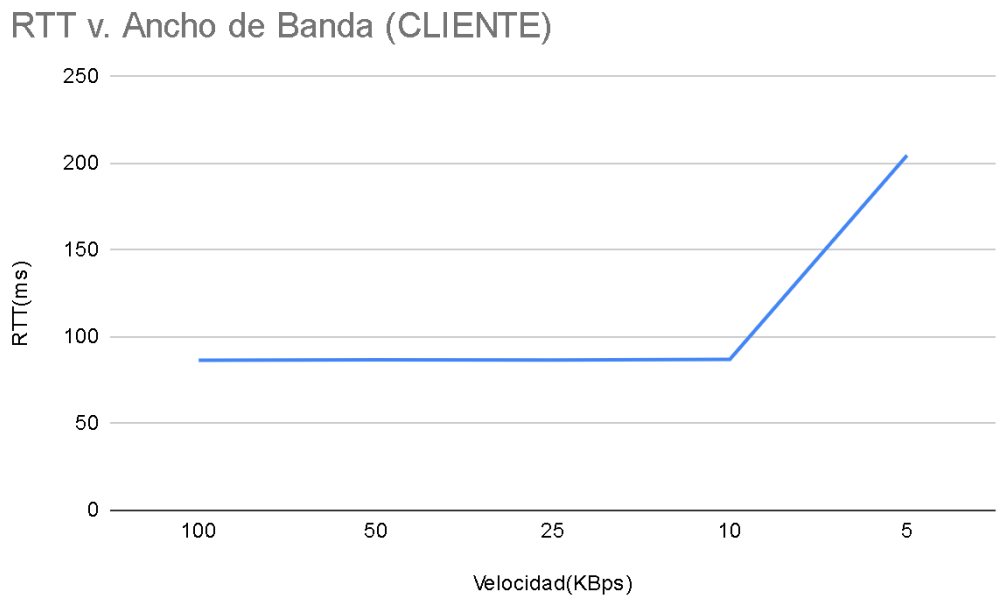

Figura 29. Resultados de control y monitoreo de robots (Cliente Web)

Discusión

Con el control y monitoreo podemos ver una historia similar que, al solo implementar el control, salvo que existió un aumento en el tiempo de respuesta al manejar un ancho de banda 5 KBps tanto en el cliente como en el robot.

Se tuvo un rango de RTT de 94 a 126 ms en el robot y un rango de 87 a 205 ms en el cliente. Esto data que el tiempo de retraso entre comando o entre recepción de dato de monitoreo puede variar entre 181 a 331 ms, donde los valores más altos pertenecen a aquellas conexiones con bajo ancho de banda.

4.5.3 Resultados de pruebas basadas en intensidad de señal

4.5.3.1 Configuración e inicialización de robots

Al momento de probar la configuración e inicialización de los robots, se lo hizo en 3 condiciones de intensidad de señal diferentes que se pueden describir como buena

intensidad de señal, señal de intensidad media y baja intensidad de señal. Estos valores se encuentran en los rangos de -30 a -50 dbm, -50 a -70 dbm, -70 a -90 dbm.

Las pruebas realizadas en el robot dieron los resultados presentes en la **tabla 9**.

Tabla 9

Resultados de configuración e inicialización de robots a base de la señal

Intensidad de Señal	**-30 a -50 dbm**	**-50 a -70 dbm**	**-70 a -90 dbm**
AVG RTT A->B (ms)	0.49395	0.6575	0.6719
AVG RTT B->A (ms)	93.29060	88.6053	95.968
AVG Packet Size A->B (bytes)	141.63	141.63	141.63
AVG Packet Size B->A (bytes)	815.93	846.15	815.93
AVG Message Time (sec)	0.72045	0.72385	7.2258

A base de esto se determinó los gráficos para resultados de configuración e inicialización de robots tanto en RTT como en duración del mensaje, estos gráficos se encuentran en la **figura 30** y la **figura 31**.

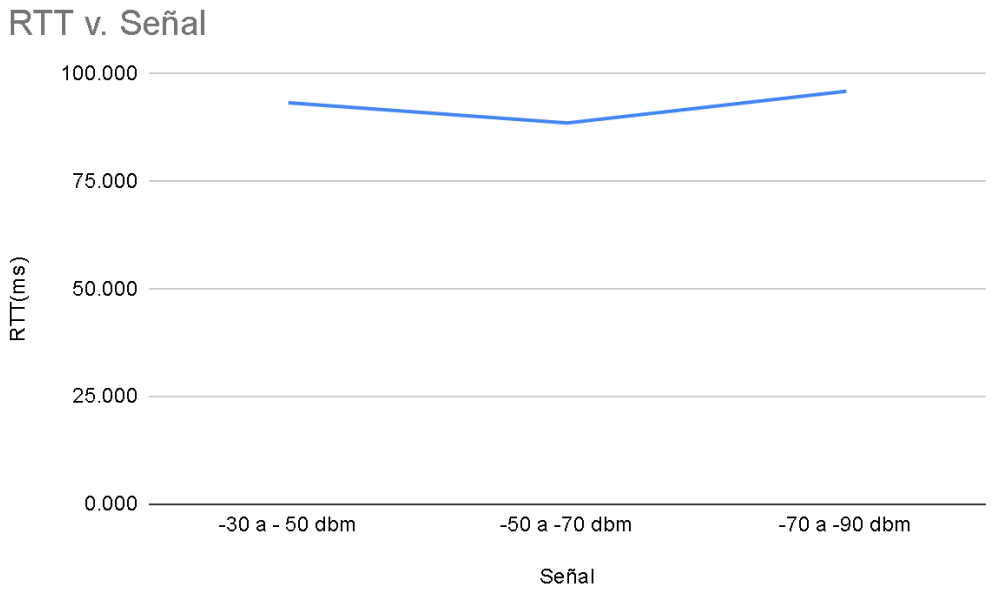

Figura 30. Resultados de configuración e inicialización de robots a base de la señal (RTT)

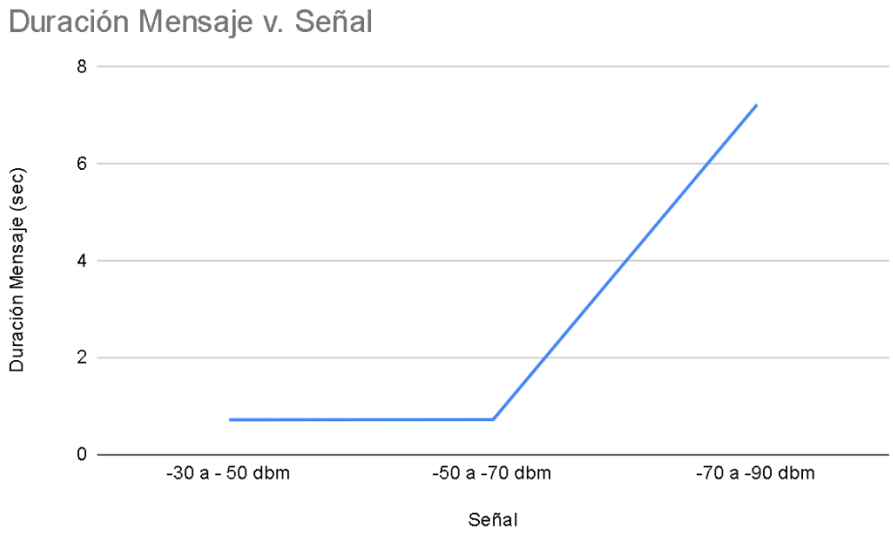

Figura 31. Resultados de configuración e inicialización de robots a base de la señal (RTT)

Discusión

Los RTT y por tanto el rendimiento general de la red no presentan mayores cambios al momento de cambiar la señal de la red, pero el tiempo general del mensaje crece en gran medida cuando se mantiene una señal entre los -70 a -90 dbm. Esto se debe principalmente a pérdidas de conexión.

4.5.3.2 Control y monitoreo de robots

Al momento de probar el control y monitoreo, se lo hizo en 3 condiciones de intensidad de señal diferentes que se pueden describir como buena intensidad de señal, señal de intensidad media y baja intensidad de señal. Estos valores se encuentran en los rangos de -30 a -50 dbm, -50 a -70 dbm, -70 a -90 dbm. Cabe destacar que los cambios en la señal solo se aplican en la red en la que se encuentran los robots. El cliente web siempre se mantiene estático.

Las pruebas realizadas en el robot dieron los resultados presentes en la **tabla 10**.

Tabla 10

Resultados de control de robots a base de la señal

Intensidad de Señal	**-30 a -50 dbm**	**-50 a -70 dbm**	**-70 a -90 dbm**
AVG RTT A->B (ms)	0.857859	1.0343	1.0706
AVG RTT B->A (ms)	89.862800	97.6760	321.7020
AVG Packet Size A->B (bytes)	78.87	83.04	82.53
AVG Packet Size B->A (bytes)	185.71	200.00	185.19

A base de esto se determinó un gráfico de análisis de RTT con la intensidad de señal en el robot que se visualiza en la **figura 32**.

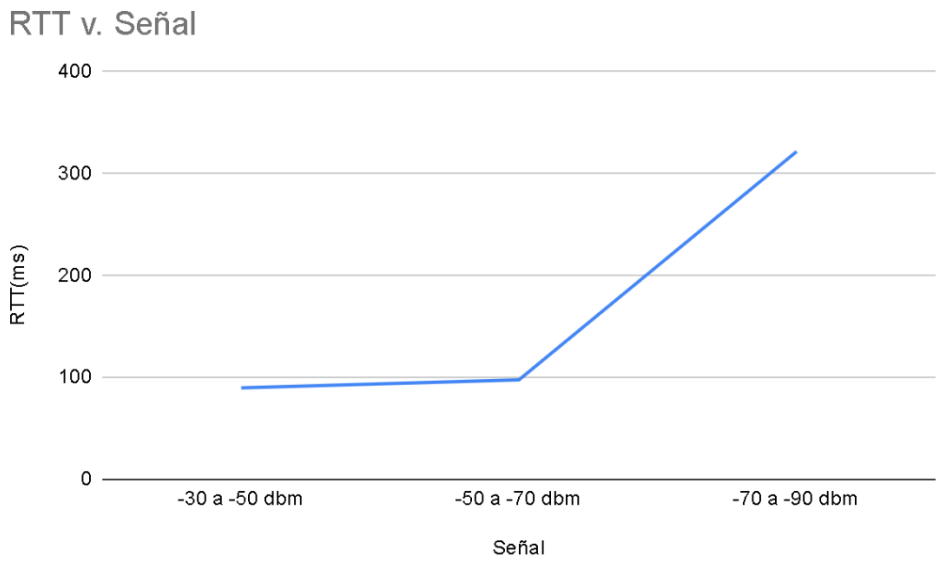

Figura 32. Resultados de control de robots a base de la señal

Discusión

Al manejar los robots en diferentes entornos de señal de red, se tiene un crecimiento considerable cuando la señal está entre los -70 a -90 dbm llegando a un RTT de 321 ms. Esto tomando el valor máximo de RTT en una conexión del cliente web en el control (89 ms) generaría un retraso en la recepción de comandos de 410 ms.

4.5.4 Rendimiento de video streaming

Aunque el video streaming no es un elemento que la arquitectura en sí, se deicidio establecer cual es un funcionamiento en el entorno y las herramientas usadas. El video streaming se lo realizó a través de YouTube en 480p. Esta prueba se lo realizó en un

entorno con 5000KBps y 1000KBps, que fueron un ancho de banda con el mejor rendimiento.

La comunicación para realizar el video streaming resultó exitosa, existió una comunicación clara entre el servidor rtmp y el robot. Pero esto no quiere decir que se puedo establecer un video continuo. El robot en su mayor parte entregaba en su mayor parte paquetes de audio y rara vez entregaba paquetes de video. Esto provocó un video intermitente y a medida que avanza el stream, el rendimiento bajó y más problemas surgían con el video. Sin importar el ancho de banda, el mismo comportamiento existió al momento de realizar streaming de video, por lo tanto, gran parte de los problemas de transmisión surgieron con el rendimiento del robot.

4.5.5 Rendimiento de servicios en la nube

4.5.5.1 Firebase (Base de datos en tiempo real)

La base de datos en tiempo es el elemento que presenta la mayor cantidad de uso dentro de toda la arquitectura y es el elemento esencial para el control y funcionamiento correcto de todo el sistema. Dentro de los datos que podemos determinar en el uso de esta base de datos tenemos su almacenamiento, sus descargas y su carga. A base de esto podemos determinar la cantidad de conexiones que se pueden establecer y los costos que pueden surgir en alguna operación. Se puede observar su uso en la **figura 33.**

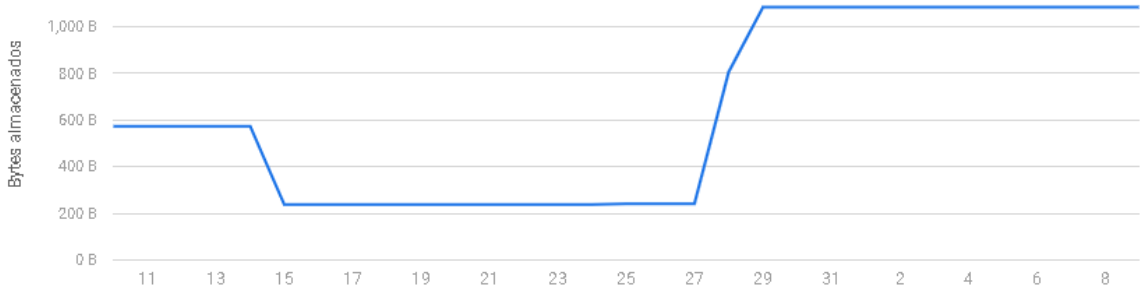

Figura 33. Bytes almacenados al largo del tiempo en la base de datos en tiempo real.

El valor máximo que se tiene en almacenamiento en la base de datos en tiempo real es de 1.1KBs y esto equivale a 11 colecciones de estructura similar y bastante simples. Cada una equivaldría a 100 bytes y cada una presenta de 2 a 3 campos de tipo string lo que daría a cada campo un valor de 30 a 50 bytes aproximadamente. Esta base de datos tiene un costo de $5 por GB almacenado (Google, n.d.-g) lo que permitiría algo similar a 1*10^6 colecciones similares a las presentes en la base de datos usada por cada dólar. Se puede observar su uso en la **figura 34**.

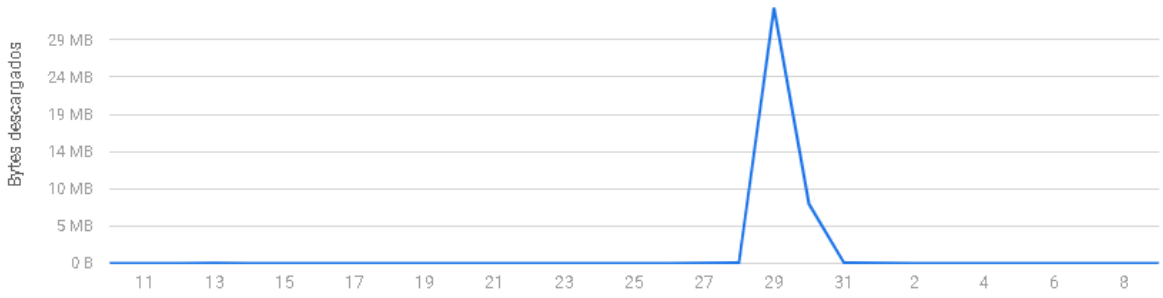

Figura 34. Bytes descargados de la base de datos en tiempo real a lo largo del tiempo.

A lo largo de las pruebas se descargó 40.7 MB, siendo el cliente web el mayor consumidor de los datos. El uso más alto consecutivo en un día fue de 32.7 MB que se lo realizó en un periodo de 3 horas de uso constante. El servicio tiene un costo de $1 por GB descargado (Google, n.d.-g). Una mayor complejidad de datos y elementos usados demandará más bytes descargados. Se puede observar su uso en la **figura 35**.

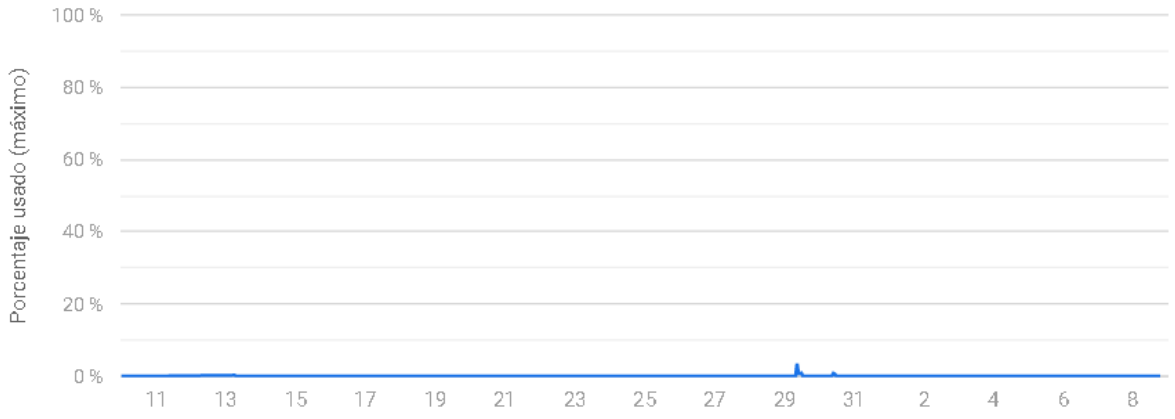

Figura 35. Porcentaje de uso de la base de datos en tiempo real a lo largo del tiempo.

El porcentaje de uso máximo fue de un 3% durante el periodo de pruebas que presentó 5 conexiones simultáneas a la base de datos. En las pruebas de maneja un máximo de 2 robots por lo tanto cada robot con su cliente web equivale a 2 conexiones y una conexión con el cliente de firebase. Se puede estimar que la máxima cantidad de conexiones antes de llegar a un 100% de uso es de 167 conexiones, lo que equivaldría a 83 robots conectados de manera simultánea. Aunque dependiendo de la carga de la conexión se puede llegar a una mayor cantidad. La cantidad máxima permitida por el servicio de firebase es de 200.000 conexiones simultáneas (Google, n.d.-g).

4.5.5.2 Firestore

Firestore en la arquitectura actúa como un almacenamiento de robots registrados, su forma de control y el historial de sus acciones. Su uso y rendimiento no es crítico para el control en tiempo de real de los robos, pero aun así es altamente usada durante su operación. Los datos que podemos determinar de su uso durante las pruebas son sus Lecturas y Escrituras. De esta manera se pueden determinar costos en su uso. Se puede observar su uso en la **figura 36**.

Figura 36. Lecturas en Firestore a lo largo del tiempo.

Las pruebas realizadas fueron el 28, 29 y 31, entre esos períodos las lecturas aumentado de 3.2 K a 4.1 K, que equivale a 900 lecturas. Las lecturas se dan principalmente al llamar a las tablas propias de la base de datos en el cliente web. 100 K lecturas equivalen a un costo de $0.06 (Google, n.d.-g). Se puede observar su uso en la **figura 37**.

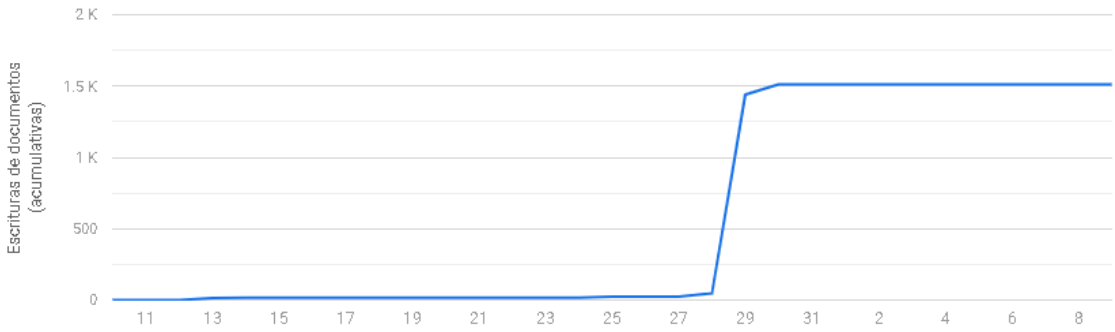

Figura 37. Escrituras en Firestore a lo largo del tiempo.

Durante el periodo de pruebas se aumentaron las escrituras de 23 a 1.5 K un aumento de más de 1 K de escrituras. Estas se dan al momento de generar los historiales de acciones de cada robot. Cada 100 K escrituras tiene un costo de $0.18.

4.5.5.3 Cloud Functions

Cloud functions son usados para levantar endpoints que permitan el registro y para activar eventos cuando se realice el control y monitoreo en tiempo real. Son usados principalmente para almacenar datos en Firestore. El dato que se puede obtener son sus invocaciones y a base de esto determinar su costo. Se puede observar esta información en la **figura 38**.

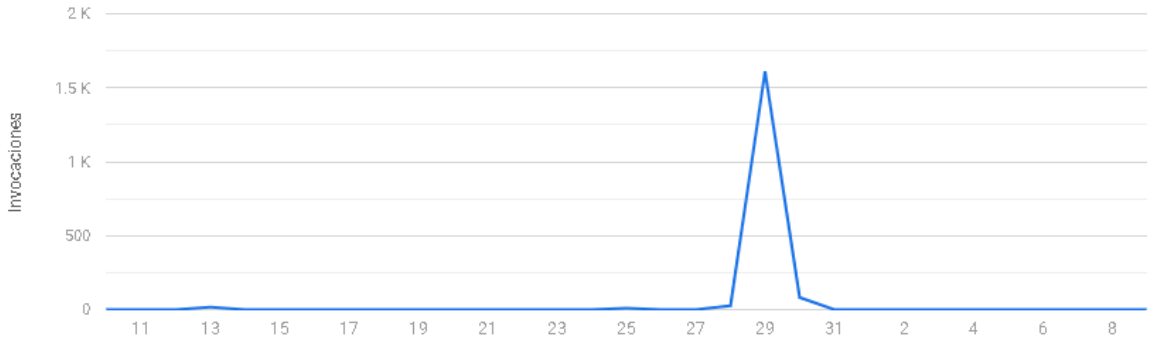

Figura 38. Invocaciones de funciones a lo largo del tiempo.

Durante las pruebas se puede establecer un total de 1.7 K invocaciones a funciones aproximadamente. Estas invocaciones se refieren al registro de robots y al registro de los historiales de acciones de cada robot. Mientras más robots existan existirá un mayor aumento de invocaciones. Cada millón de invocaciones equivalen a $0.40 (Google, n.d.-g).

CAPÍTULO V

ANÁLISIS E INTERPRETACIÓN DE RESULTADOS

5.1 INTRODUCCIÓN

Una vez establecida la arquitectura para el control y monitoreo de robots de búsqueda y rescate y realizadas las pruebas; fue necesario llevar a cabo un análisis detallado de los resultados obtenidos. Este análisis permitió entender en qué aspectos la arquitectura aporta con el control y monitoreo de robots de búsqueda y rescate urbano, y en qué aspectos se presentan diferentes limitaciones.

Gracias al análisis e interpretación de los resultados obtenidos de las pruebas se determinó el rendimiento de la arquitectura en la configuración, en el control y el monitoreo de robots tomando en cuenta el ancho de banda y la calidad de señal en la red. Además de eso, se determinó la capacidad de realizar streaming de video al momento de control.

Con todo esto, se llevó a cabo la determinación de qué elementos de la arquitectura son útiles para operaciones de búsqueda y rescate. Se establece elementos como escalabilidad, adaptabilidad, utilidad, entre otros. De allí, también se pudo encontrar las diferentes limitantes que presenta la arquitectura en su estructura propuesta y las acciones que se deberían llevar a cabo para contrarrestar estas limitaciones.

5.2 ANÁLISIS DE RESULTADOS

5.2.1 Análisis de la configuración de los robots

La configuración de robots resultó en un proceso bastante simple y con un alto rendimiento en la red, pero que se encarga de enviar una cantidad considerable de datos a la nube para establecer las características del robot, por lo que es dependiente en parte del ancho de banda en la red. Una vez estando en una velocidad de 10KBps se empezó a observar retrasos en el tiempo de carga de la información, y a 5KBps este tiempo aumentó aún más. Se puede esperar que configuraciones complejas requieren de mayor ancho de banda y por lo tanto es preferible que este proceso se realice en redes de alta capacidad, aunque puede funcionar sin mayor problema con un ancho de banda reducida.

Aunque el ancho de banda no presenta un mayor problema con el proceso, es absolutamente necesario que la señal de la red sea de calidad. En el caso de las pruebas, la conexión que fue realizada por wifi, cualquier petición en una red con baja señal podría resultar en errores y tiempos para completar configuración algo considerable.

5.2.2 Análisis de control y monitoreo

El control y monitoreo funciona de manera adecuada en entornos de ancho de banda reducido, sin mayores retrasos ni problemas, por lo tanto, se puede controlar y recibir información de los robots sin mayores fluctuaciones en la latencia; incluso a 5KBps. Claro está que esto se mantiene mientras que la complejidad de datos a enviar no sea muy grande. Los datos por enviar siempre deben mantenerse en un paquete TCP y el tamaño del paquete debe ser bastante reducido con relación al ancho de banda usado. Los comandos de control por lo tanto no deben ser demasiado complejos ni extensos, y lo mismo aplica para los datos de monitoreo enviados por los robots. En la prueba se tuvo paquetes que tenían una variación entre los 100 a 300 bytes.

Una comunicación ideal estableció una comunicación con latencias totales que oscilan en los 200ms y en el peor de los casos podían llegar a 300ms. Esta latencia toma en cuenta la latencia de comunicación del robot con la nube y del cliente web con la nube. Individualmente esta latencia oscila entre los 90 a 100 ms y es altamente dependiente de la latencia que se tienen con los servicios de la nube. Es decir que, si nos encontráramos a una distancia más cercana del lugar de alojamiento de los servicios, menor seria la latencia y mejor el control.

Una latencia de 200ms aunque algo que se puede percibir claramente, no es de una tamaño demasiado considerable para evitar un control efectivo (Deber, Jota,

Forlines, & Wigdor, 2015; Kämäräinen, Siekkinen, Ylä-Jääski, Zhang, & Hui, 2017). Además de esto, la arquitectura no sobrepasa el nivel de latencia recomendado por (ITU, 2003) de 400ms en comunicaciones de red.

Todos estos valores son útiles mientras la señal de red no sea de baja calidad. Una señal de baja calidad, aunque no afecta mucho el tiempo de respuesta en el envío de cada paquete; genera más errores y produce mayores retrasos afectando la funcionalidad de la arquitectura y evitando un control y monitoreo adecuado.

5.2.3 Análisis con el video streaming

El video streaming es un elemento que no es del todo adaptable en la arquitectura, al momento de usarla se presentaron una cantidad extensa de errores, retrasos y una latencia alta. Esto se debió principalmente que su implementación por medio del internet resulta una actividad que requiere de alta cantidad de recursos de parte del robot y una excelente calidad de señal y ancho de banda en la red. Otras alternativas o métodos son necesarios.

5.2.4 Análisis de servicios en la nube

El funcionamiento de la arquitectura en la nube permite un despliegue sencillo, rápido y bastante adaptable bajo un modelo serverless. Claro está, que se dependiente de los servicios usados (firebase) y del proveedor de los servicios (google). De todas formas, provee un entorno simple y de alto rendimiento. Además de esto, el costo de usar estos servicios se basa en el uso de estos. Esto quiere decir que mientras más tiempo y más compleja una operación de búsqueda y rescate más costosa será.

El principal costo de la arquitectura vendría de los servicios en la nube y la otra parte del diseño de los robots.

5.3 UTILIDAD EN OPERACIONES DE BÚSQUEDA Y RESCATE URBANO

La arquitectura diseñada busca que sea posible el control y monitoreo de múltiples robots a través de la nube en un entorno serverless y que así pueda cumplir con diferentes procesos propios de operaciones de búsqueda y rescate urbano. Más específicamente cumplir con las fases propuestas por INSARAG: Preparación, Movilización, Operaciones, Desmovilización y Post Misión (INSARAG, n.d.-a).

La arquitectura con su diseño funciona y rinde adecuadamente en las pruebas establecidas y presenta diferentes elementos que le dan un gran potencial de utilidad en la búsqueda y rescate urbano. Su latencia de control ronda en los 200 ms, mientras que es algo notable, no lo hace difícil de usar. Aunque su uso en streaming de video es considerablemente limitado y presenta limitaciones.

La arquitectura es fácilmente adaptable y flexible debido a que permite una gran variedad de robots, permite alta escalabilidad en su arquitectura y no requiere de mayor hardware para funcionar. La arquitectura presenta un cliente web, pero esto no quiere decir que solo puede funcionar en entornos web. También y fácilmente puede ser usado en móviles o en aplicaciones especializadas que pueden conectarse a la arquitectura. Además de eso, esta puede cambiar y de manera sencilla puede recibir nuevos elementos gracias a su naturaleza serverless y su carencia de elementos físicos.

La arquitectura es escalable, debido a que puede funcionar en entornos con una gran cantidad de robots y dispositivos conectados y continuar con rendimiento alto.

Aunque claro está que presenta límites antes que se requieran establecer medidas para expandir la capacidad de la arquitectura.

Los robots pueden integrarse de manera rápida a la arquitectura y pueden tener diferentes roles. La arquitectura no tiene mayores limitantes para que pueda funcionar con una variedad de robots, sean terrestres o no. El limitante es que estos robots tienen que ajustarse al modelo de arquitectura. Es decir, que deben poder conectarse con los servicios de nube usados (Firebase) y deben tener un archivo de configuración y mapeo de control que se ajuste a su funcionalidad. Con esto, puede funcionar libremente dentro de todo el sistema.

5.4 LIMITACIONES

La arquitectura planteada como tal y con el entorno probado, aunque presenta una funcionalidad y utilidad adecuada, también tiene una serie de limitaciones para uso en el campo y en gran medida.

La arquitectura como tal, actualmente solo funciona con robots que tengan una conexión a internet sea por WiFi o cableado. Su conexión a internet debe de ser estable y de alta calidad, aunque no requiere de un ancho de banda considerable. Con 5KBps funciona adecuadamente. Aun así, el hecho de necesitar de una de una señal WiFi o similar limita el rango sobre los que robots pueden ser usados y por lo tanto otros medios de conexión son necesarios.

Streaming de video o entrega de imagen, que es un elemento sumamente útil para poder usar robots de manera remota, es un elemento que la arquitectura no soporta adecuadamente. Requiere de un alto ancho de banda y un poder de hardware

considerable para funcionar. Es necesario elementos de streaming externos para un funcionamiento adecuado.

La arquitectura debido a su naturaleza serverless, que le permite facilidad de configuración e implementación, también limita el uso a los servicios del proveedor (en este caso Google). Esto puede llegar a ser un limitante para crear configuraciones sumamente específicas que no soporte el proveedor.

La arquitectura funciona de manera conceptual y ha demostrado tener un rendimiento adecuado manteniendo un gran nivel de funcionalidad, versatilidad, escalabilidad y simplicidad. Aun así, es necesario llevar a aumentar el nivel de pruebas y asegurar que esta arquitectura puede llegar a funcionar en ambientes reales de búsqueda y rescate de una manera efectiva y adecuada.

CAPÍTULO VI

CONCLUSIONES Y LÍNEAS DE TRABAJO FUTURO

6.1 CONCLUSIONES

El presente trabajo procedió a establecer una arquitectura de control y monitoreo para robots de búsqueda y rescate urbano que se presenta de una manera simple en su complejidad técnica y de uso, procurando la utilidad para operaciones de búsqueda y rescate. Todo esto con el uso de tecnologías serverless de la nube como un elemento fundamental para asegurar la utilidad de la arquitectura. Para lograr todo esto, se

determinó los diferentes elementos que existen en las operaciones de búsqueda y rescate urbano actuales y sus relaciones con los robots usados y también con los servicios de la nube que existen. Así, se hizo uso de la especificación dada por la INSARAG para llevar a cabo una operación de búsqueda y rescate como una base de la funcionalidad. Con eso se pudo establecer que es necesario seguir fases de preparación, movilización, operaciones, desmovilización y post -mision. De allí se pudo establecer una arquitectura y sus procesos que pueda ser fundamentalmente útiles en la búsqueda y rescate urbanos. Esta arquitectura, permitía en general, registrar nuevos robots a base de especificación de control y funcionamiento, controlar a los robots, recibir datos de monitoreo de los robots; todo estos a través de servicios serverless y un cliente web. El funcionamiento de esta arquitectura se basó en el uso de Firebase como la fuente principal de las herramientas serverless. Los robots podían acoplarse a la arquitectura siempre y cuando pudieran comunicarse con Firebase y presentaran una estructura funcional y un mapa de control. Aunque con una estructura y funcionalidad que cumplía con el objetivo a lograr, necesitaba un proceso de validación que le dieran un indicio de utilidad. Las pruebas realizadas fueron ejecutadas en un entorno controlado donde se pueda alterar la intensidad de señal y el ancho de banda de la comunicación de los robots. Estas pruebas determinaron que los procesos de configuración, control y monitoreo son adecuados y útiles, aunque se presentan latencias perceptibles, pero que no invalidan la arquitectura. También se encontró que estas latencias son dependientes en gran medida de la distancia de los servicios con los clientes de la arquitectura (usuario y robots), y por lo tanto es necesario un posicionamiento inteligente para reducir retrasos y latencias. Además de esto, se encontró que la arquitectura puede funcionar en entornos de bajo

ancho de banda pero que requieren buena señal de conexión. Fuera de esto, se encontraron diversas limitaciones, entre estas; la dependencia de una conexión wifi limita el rango de utilidad de los robots y el uso de video streaming requiere de elementos externos a la arquitectura para funcionar efectivamente. Los costos de uso de la arquitectura son extremadamente flexibles y variables al uso que se le dé y no es posible generar una estimación certera sin pruebas en un entorno real.

6.2 LÍNEAS DE TRABAJO FUTURO

Es necesario encontrar medios de conexiones alternativos a WiFi para las comunicaciones pero que a su vez permitan conexiones con los servicios de la nube usados para mantener el mínimo de latencias.

El video streaming debe ser implementado de una manera formal en la arquitectura. Este elemento debe ser funcional y efectivo para una operación de búsqueda y rescate. No es necesario que este haga uso de tecnologías serverless o se ha altamente dependiente del internet, pero si se debe tomar en cuenta su rango de uso y versatilidad de uso en diversos dispositivos.

Aunque la arquitectura en sus pruebas de QoS entregó resultados positivos, es necesario que se proceda a realizar estudio de la arquitectura en entornos similares a la realidad. De esta manera se podrá determinar cambios necesarios para asegurar el mejor funcionamiento en una operación real, el uso óptimo de los robots y su construcción, y por último el costo real del uso de la arquitectura.

BIBLIOGRAFÍA

Alex, C., & Vijaychandra, A. (2017). Autonomous cloud based drone system for disaster response and mitigation. *International Conference on Robotics and Automation for Humanitarian Applications, RAHA 2016 - Conference Proceedings*, 1–4. https://doi.org/10.1109/RAHA.2016.7931889

Baldini, I., Castro, P., Chang, K., Cheng, P., Fink, S., Ishakian, V., … Suter, P. (2017). Serverless Computing: Current Trends and Open Problems. In *Research Advances in Cloud Computing* (pp. 1–20). https://doi.org/10.1007/978-981-10-5026-8_1

Buschmann, F. (2013). *Pattern-oriented software architecture. Volume 1, A system of patterns*. Wiley.

Cubber, G. De, Doroftei, D., Rudin, K., Berns, K., Matos, A., Serrano, D., … Ourevitch, S. (2017). Introduction to the Use of Robotic Tools for Search and Rescue. In *Search and Rescue Robotics - From Theory to Practice*. https://doi.org/10.5772/intechopen.69489

Deber, J., Jota, R., Forlines, C., & Wigdor, D. (2015). How much faster is fast enough? User perception of latency & latency improvements in direct and indirect touch. *Conference on Human Factors in Computing Systems - Proceedings*, *2015-April*, 1827–1836. https://doi.org/10.1145/2702123.2702300

Definition: round-trip delay time. (n.d.). Retrieved November 27, 2019, from https://www.its.bldrdoc.gov/fs-1037/dir-031/_4641.htm

Dey, S., Bhattacharyya, A., & Mukherjee, A. (2017). Semantic data exchange between

collaborative robots in fog environment: Can CoAP be a choice? *GIoTS 2017 - Global Internet of Things Summit, Proceedings*. https://doi.org/10.1109/GIOTS.2017.8016232

Firebase – Firebase console. (n.d.). Retrieved November 26, 2019, from https://console.firebase.google.com/u/0/project/tsrrcontrolmonit/overview

Gharibi, M., Boutaba, R., & Waslander, S. L. (2016). Internet of Drones. *IEEE Access*, *4*(JANUARY), 1148–1162. https://doi.org/10.1109/ACCESS.2016.2537208

Google. (n.d.-a). Android Developers Blog: Android Things Developer Preview 6. Retrieved November 26, 2019, from https://android-developers.googleblog.com/2017/11/android-things-developer-preview-6.html

Google. (n.d.-b). Choose a database: Cloud Firestore or Realtime Database | Firebase. Retrieved November 26, 2019, from https://firebase.google.com/docs/firestore/rtdb-vs-firestore

Google. (n.d.-c). Cloud Firestore | Firebase. Retrieved November 26, 2019, from https://firebase.google.com/docs/firestore

Google. (n.d.-d). Cloud Functions for Firebase | Firebase. Retrieved November 26, 2019, from https://firebase.google.com/docs/functions

Google. (n.d.-e). Documentation | Firebase. Retrieved November 26, 2019, from https://firebase.google.com/docs

Google. (n.d.-f). Firebase Hosting | Firebase. Retrieved November 26, 2019, from

https://firebase.google.com/docs/hosting

Google. (n.d.-g). Firebase Pricing | Firebase. Retrieved December 2, 2019, from https://firebase.google.com/pricing/?hl=es

Google. (n.d.-h). Firebase Realtime Database. Retrieved November 26, 2019, from https://firebase.google.com/docs/database/

Google. (n.d.-i). Kotlin overview | Desarrolladores de Android | Android Developers. Retrieved November 26, 2019, from https://developer.android.com/kotlin/overview?hl=es

Google. (n.d.-j). Overview | Android Things | Android Developers. Retrieved November 26, 2019, from https://developer.android.com/things/get-started/

Goverment of Canada. (2018). About Search and Rescue (SAR).

Heiss, J. J. (2013). The Advent of Kotlin: A Conversation with JetBrains' Andrey Breslav. Retrieved November 26, 2019, from https://www.oracle.com/technical-resources/articles/java/breslav.html

ICARUS. (n.d.). Search & Rescue.

INSARAG. (n.d.-a). *Guías de INSARAG. Volumen II: Manual B; Operaciones. OCHA.*

INSARAG. (n.d.-b). *Guías de INSARAG Volumen II: Manual C; Clasificación y Reclasificación Externa de INSARAG*.

INSARAG. (n.d.-c). INICIO.

ITU. (2003). Tiempo de transmisión en un sentido, Recomendación UIT-T G.114. *Uit-T*.

Kämäräinen, T., Siekkinen, M., Ylä-Jääski, A., Zhang, W., & Hui, P. (2017). A measurement study on achieving imperceptible latency in mobile cloud gaming. *Proceedings of the 8th ACM Multimedia Systems Conference, MMSys 2017*, 88–99. https://doi.org/10.1145/3083187.3083191

Kim, Y. D., Jung, S. H., Gu, D. Y., Kim, H. K., & Song, C. H. (2017). IoT Sensor Based Mobility Performance Test-Bed for Disaster Response Robots. *Proceedings - 2017 6th IIAI International Congress on Advanced Applied Informatics, IIAI-AAI 2017*, 990–991. https://doi.org/10.1109/IIAI-AAI.2017.32

Kitchenham, B., & Charters, S. (2007). issue: EBSE 2007-001. *Technical Report*, *2*(3).

Latif, T., Whitmire, E., Novak, T., & Bozkurt, A. (2016). Sound Localization Sensors for Search and Rescue Biobots. *IEEE Sensors Journal*, *16*(10), 3444–3453. https://doi.org/10.1109/JSEN.2015.2477443

Lee, J., Wang, J., Crandall, D., Sabanovic, S., & Fox, G. (2017). Real-time, cloud-based object detection for unmanned aerial vehicles. *Proceedings - 2017 1st IEEE International Conference on Robotic Computing, IRC 2017*, 36–43. https://doi.org/10.1109/IRC.2017.77

Marston, T. (2012). What is the 3-Tier Architecture? Retrieved November 27, 2019, from http://www.tonymarston.net/php-mysql/3-tier-architecture.html#n-tier-and-3-tier

McGrath, G., & Brenner, P. R. (2017). Serverless Computing: Design, Implementation, and Performance. *2017 IEEE 37th International Conference on Distributed*

Computing Systems Workshops (ICDCSW), 405–410. https://doi.org/10.1109/ICDCSW.2017.36

Miranda, L. (n.d.). Google tira la toalla con Android Things. Retrieved November 26, 2019, from https://hipertextual.com/2019/02/google-android-things

Mouradian, C., Yangui, S., & Glitho, R. H. (2018). Robots as-a-service in cloud computing: Search and rescue in large-scale disasters case study. *CCNC 2018 - 2018 15th IEEE Annual Consumer Communications and Networking Conference*, *2018-Janua*, 1–7. https://doi.org/10.1109/CCNC.2018.8319200

Moyers, E. (2015). Why is almost everything negative in Wi... - Cisco Community. Retrieved December 2, 2019, from https://community.cisco.com/t5/small-business-support-documents/why-is-almost-everything-negative-in-wireless/ta-p/3159743

Mozilla. (n.d.). JavaScript | MDN. Retrieved November 26, 2019, from https://developer.mozilla.org/es/docs/Web/JavaScript

Murphy, R. R., Tadokoro, S., & Kleiner, A. (2016). Disaster Robotics. In *Springer Handbook of Robotics* (pp. 1577–1604). https://doi.org/10.1007/978-3-319-32552-1_60

Nakamura, K., Chen, W., Ogawa, J., Watanobe, Y., Yaguchi, Y., & Naruse, K. (2018). A Study of Robotic Cooperation in Cloud Robotics: Architecture and Challenges. *IEEE Access*, *6*, 36662–36682. https://doi.org/10.1109/access.2018.2852295

Nurmi, P., & Tarkoma, S. (2017). Low-cost support for search and rescue operations using off-the-shelf sensor technologies. *Electronics Letters*, *53*(15), 1011–1013.

https://doi.org/10.1049/el.2017.1519

Pelka, M., Majek, K., Bedkowski, J., Musialik, P., Maslowski, A., De Cubber, G., … Govindaraj, S. (2014). Training and support system in the cloud for improving the situational awareness in Search and Rescue (SAR) operations. *12th IEEE International Symposium on Safety, Security and Rescue Robotics, SSRR 2014 - Symposium Proceedings*. https://doi.org/10.1109/SSRR.2014.7017644

Petersen, K. and Feldt, R. and Mujtaba, S. and Mattsson, M. (2008). Systematic Mapping Studies in Software Engineering. *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*, (February 2015), 68–77.

Python. (n.d.). BeginnersGuide/Overview - Python Wiki. Retrieved November 26, 2019, from https://wiki.python.org/moin/BeginnersGuide/Overview

Rahman, M. A., Azad, S., Asyhari, A. T., Bhuiyan, M. Z. A., & Anwar, K. (2018). Collab-SAR: A Collaborative Avalanche Search-and-Rescue Missions Exploiting Hostile Alpine Networks. *IEEE Access*, *6*(c), 42094–42107. https://doi.org/10.1109/ACCESS.2018.2848366

RASPBERRY PI, F. (n.d.-a). Buy a Raspberry Pi 3 Model B+ – Raspberry Pi. Retrieved November 26, 2019, from https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/

RASPBERRY PI, F. (n.d.-b). FrontPage - Raspbian. Retrieved November 26, 2019, from https://www.raspbian.org/

RASPBERRY PI, F. (n.d.-c). Raspberry Pi Documentation. Retrieved November 26, 2019, from https://www.raspberrypi.org/documentation/

RASPBERRY PI, F. (n.d.-d). Raspbian Stretch has arrived for Raspberry Pi - Raspberry Pi. Retrieved November 26, 2019, from https://www.raspberrypi.org/blog/raspbian-stretch/

RASPBERRY PI, F. (n.d.-e). What is a Raspberry Pi? Retrieved November 26, 2019, from https://www.raspberrypi.org/help/what- is-a-raspberry-pi/

Rittinghouse, J. W., & Ransome, J. F. (2017). *Cloud Computing*. https://doi.org/10.1201/9781439806814

Sadiku, M. N. O., Musa, S. M., & Momoh, O. D. (2014). Cloud Computing: Opportunities and Challenges. *IEEE Potentials*, *33*(1), 34–36. https://doi.org/10.1109/MPOT.2013.2279684

Sharma, M., Young, J. E., & Eskicioglu, R. (2012). *Developing guidelines for in-the-field control of a team of robots*. 233. https://doi.org/10.1145/2157689.2157771

TCPDUMP/LIBPCAP public repository. (n.d.). Retrieved November 26, 2019, from http://www.tcpdump.org/

tcpdump. (n.d.). *Tcpdump/Libpcap public repository*.

Wan, J., Tang, S., Yan, H., Li, D., Wang, S., & Vasilakos, A. V. (2016). Cloud robotics: Current status and open issues. *IEEE Access*, *4*, 2797–2807. https://doi.org/10.1109/ACCESS.2016.2574979

Wireshark. (n.d.). Wireshark · Go Deep. Retrieved November 26, 2019, from https://www.wireshark.org/index.html#aboutWS

Xin, C., Qiao, D., Hongjie, S., Chunhe, L., & Haikuan, Z. (2018). Design and Implementation of Debris Search and Rescue Robot System Based on Internet of Things. *Proceedings - 2018 International Conference on Smart Grid and Electrical Automation, ICSGEA 2018*, 303–307. https://doi.org/10.1109/ICSGEA.2018.00082