ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELETRÓNICA

CARRERA TELECOMUNICACIONES

DISEÑO E IMPLEMENTACIÓN DE UNA RED DE ACCESO A SERVICIOS DE REALIDAD VIRTUAL ORIENTADO AL IOT

Trabajo de Titulación

Tipo: Proyecto de Investigación

Presentado para optar al grado académico de:

INGENIERA EN TELECOMUNICACIONES

AUTORA:

CINTHYA PIEDAD HUILCAMAYGUA DE LA CRUZ

Riobamba – Ecuador

2023

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA TELECOMUNICACIONES

DISEÑO E IMPLEMENTACIÓN DE UNA RED DE ACCESO A SERVICIOS DE REALIDAD VIRTUAL ORIENTADO AL IOT

Trabajo de Titulación

Tipo: Proyecto de Investigación

Presentado para optar al grado académico de:

INGENIERA EN TELECOMUNICACIONES

AUTORA: CINTHYA PIEDAD HUILCAMAYGUA DE LA CRUZ

DIRECTOR: ING. DIEGO FERNANDO VELOZ CHERREZ MSC.

Riobamba – Ecuador

2023

Yo, Cinthya Piedad Huilcamaygua De la Cruz, declaro que el presente trabajo de titulación es de mi autoría y los resultados del mismo son auténticos. Los textos en el documento que provienen de otras fuentes están debidamente citados y referenciados.

Como autora asumo la responsabilidad legal y académica de los contenidos de este trabajo de titulación. El patrimonio intelectual pertenece a la Escuela Superior Politécnica de Chimborazo.

Riobamba, 13 de diciembre de 2023

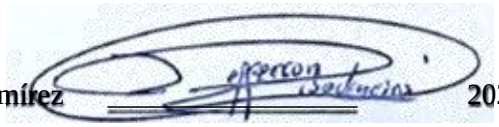
Cinthya Piedad Huilcamaygua De la Cruz
172495075-1

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA TELECOMUNICACIONES

El Tribunal del Trabajo de Titulación certifica que: El Trabajo de Titulación; tipo: Proyecto de Investigación, **DISEÑO E IMPLEMENTACIÓN DE UNA RED DE ACCESO A SERVICIOS DE REALIDAD VIRTUAL ORIENTADO AL IOT**, realizado por la señorita: **CINTHYA PIEDAD HUILCAMAYGUA DE LA CRUZ**, ha sido minuciosamente revisado por los Miembros del Tribunal del Trabajo de Titulación, el mismo que cumple con los requisitos científicos, técnicos, legales, en tal virtud el Tribunal Autoriza su presentación.

	FIRMA	**FECHA**
Ing. Marco Vinicio Ramos Valencia **PRESIDENTE DEL TRIBUNAL**		2023-12-13
Ing. Diego Fernando Veloz Chérrez **DIRECTOR DEL TRABAJO DE TITULACIÓN**		2023-12-13
Ing. Jefferson Alexander Ribadeneira Ramírez **ASESOR DEL TRABAJO DE TITULACIÓN**		2023-12-13

DEDICATORIA

Este trabajo de investigación se lo dedico en primer lugar a Dios, por haberme guiado en todos estos años de preparación universitaria y a fortalecerme con cada obstáculo en el camino. A mis padres, Blanca y Gerardo, quienes siempre han sido el motor constante para lograr este cometido y que desde siempre me han inculcado que la educación es la mejor herencia que se puede brindar a un hijo. A David, mi amado esposo, que me ha impulsado a perseguir mis sueños y a no rendirme ante las dificultades, su apoyo incondicional y paciencia han sido fundamentales para alcanzar este logro. A mis hermanas, y en especial a Myrian por ser mi fuente de motivación y quien en ella vi un ejemplo a seguir. A todos ustedes, les dedico esta tesis con todo mi corazón.

Cinthya

AGRADECIMIENTO

Quiero expresar mi sincero agradecimiento a todas las personas que han contribuido a hacer posible este sueño. En especial, agradezco a mis profesores y tutores de la prestigiosa Escuela Superior Politécnica de Chimborazo, cuya guía y conocimiento han sido fundamentales para mi formación y para el desarrollo de esta investigación. También quiero agradecer a mis amigos y familiares, por su amor, apoyo y comprensión durante este proceso. Este logro no hubiera sido posible sin su ayuda y sin el constante aliento de todos aquellos que me han acompañado en este camino.

Cinthya

ÍNDICE DE CONTENIDO

CAPÍTULO II

CAPÍTULO III

ÍNDICE DE TABLAS

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE ANEXOS

RESUMEN

El presente trabajo de investigación ha buscado diseñar e implementar una red de acceso a servicios de realidad virtual orientado al Internet de las cosas (IoT), se abordó el problema sobre si la eficiencia en el acceso a servicios de realidad virtual (RV) que podría ser mejorada al implementar una red orientada al IoT. Con el propósito de evaluar el rendimiento de la red, este estudio empleó una metodología cuantitativa. Primero, se indagó en la literatura actual sobre protocolos e infraestructuras IoT, lo que permitió identificar protocolos clave y comprender su idoneidad en relación con otros. Posteriormente, se trazó y estableció la estructura de la red, enfocándose en satisfacer requisitos específicos del acceso a servicios de RV. En la evaluación, se contrastó el uso del protocolo MQTT (Message Queuing Telemetry Transport) y el protocolo HTTP en dos sistemas separados, ambos sistemas emplean la misma estructura de red, pero con programaciones para funcionar con el respectivo protocolo. Además, para solventar las interacciones en tiempo real de HTTP se empleó WebSocket. En este contexto, se recolectaron parámetros de rendimiento utilizando ambos protocolos en el entorno de la realidad virtual. Los resultados mostraron que MQTT presentaba una superioridad en términos generales en comparación con HTTP bajo las mismas condiciones, pero tendría una gran similitud con WebSocket. Para enviar datos puntuales, MQTT es mejor que HTTP en un 42% para el uso de ancho de banda; un 2.61% en tiempos de RTT; tiene porcentajes de error cercanos a cero a diferencia de HTTP que tiene un porcentaje de error del 2% de errores. En conclusión, para aplicaciones de RV que requieren comunicación en tiempo real y eficiencia en la transmisión, los protocolos MQTT y WebSocket se presentan como una opción preferible a comparación de HTTP.

Palabras clave: <INTERNET DE LAS COSAS (IOT)>, <REALIDAD VIRTUAL (VR)>, <COMUNICACIÓN EN TIEMPO REAL>, <PROTOCOLO MQTT>, <PROTOCOLO HTTP>, <EVALUACIÓN>, <UNITY (SOFTWARE)>, <AGENTE INTERMEDIO (BROKER)>.

SUMMARY

The objective of this research was to develop and implement a network for accessing virtual reality services with a focus on the Internet of Things (IoT). The study addressed the question of whether implementing an IoT-oriented network could enhance the efficiency of accessing virtual reality (VR) services. The Quantitative methodology was employed to assess the network's performance. Initially, a literature review of IoT protocols and infrastructures identified and understood their suitability relative to others. Following that, the network structure was outlined and established, emphasizing on meeting the requirements for accessing VR services. In the evaluation, the MQTT performance (Message Queuing Telemetry Transport) protocol was contrasted with the HTTP protocol in two separate systems. Both systems shared the same network structure but were programmed to operate with their respective protocols. Additionally, WebSocket addresses real-time interactions in the HTTP system. Performance parameters were gathered using both protocols in the virtual reality environment. The results demonstrated that MQTT exhibited superior overall performance compared to HTTP under similar conditions, although it showed notable similarity to WebSocket. Concerning sending specific data points, MQTT surpassed HTTP by 42% in bandwidth usage, 2.61% in Round-Trip Times (RTT), and demonstrated error percentages close to zero. In contrast, HTTP exhibited a 2% error rate. In conclusion, for VR applications necessitating real-time communication and transmission efficiency, MQTT and WebSocket protocols emerge as preferable options over HTTP.

KEYWORDS: <INTERNET OF THINGS (IOT)>, <VIRTUAL REALITY (VR)>, <REAL-TIME COMMUNICATION>, <MQTT PROTOCOL>, <HTTP PROTOCOL>, <EVALUATION>, <UNITY (SOFTWARE)>, <BROKER AGENT>.

Lic. Maritza Larrea Mg.
0603370784

INTRODUCCIÓN

En la era digital actual, el auge del Internet de las Cosas (IoT) ha revolucionado la forma en que los dispositivos se comunican entre sí y cómo interactúan con el entorno. Esta revolución no solo ha influido en campos tradicionales como la industria y la domótica, sino que también ha dejado su huella en sectores emergentes, como la realidad virtual (RV).

A medida que surgen distintas modalidades de comunicación y se debaten las ventajas y limitaciones de cada una, se torna imperativo analizar cómo estas modalidades influyen en eficiencia de la red para servicios de RV, especialmente cuando se incorpora al ámbito del IoT. No es solo una cuestión de cómo se conectan los dispositivos, sino de cómo esa conexión mejora o limita la experiencia del usuario en un entorno de RV.

En el presente trabajo de investigación, se pretende emplear un protocolo de comunicación IoT con el propósito de facilitar el envío de mensajes, transformando así una aplicación de realidad virtual en un elemento IoT que emite información hacia un bróker alojado en un servidor en la nube. En este estudio, se plantea un cambio de paradigma, desafiando la creencia de que los protocolos de comunicación de Internet de las Cosas (IoT) deben restringirse únicamente al uso en sistemas donde solo se involucren objetos controlados. Se demuestra que estos protocolos también pueden ser empleados para otros fines, como el acceso a servicios de realidad virtual y entorno multiusuario.

Con este panorama, se pretende adentrarse en una exploración profunda, evaluando aspectos como los parámetros de rendimiento que involucran el tiempo de ida y vuelta de los paquetes, el ancho de banda utilizado y la perdida de información. Para ello, se seguirá un enfoque estructurado que no solo se basa en teorías existentes, sino que también incorpora pruebas prácticas y observaciones, proporcionando así una comprensión integral y revelando áreas para futuras investigaciones y desarrollos en este confluente de RV e IoT.

CAPÍTULO I

1. PROBLEMA DE INVESTIGACIÓN

1.1 Planteamiento del problema

Desde los primeros indicios del ordenador y la posterior aparición de las redes, las personas han experimentado cambios en la forma de comunicarse. Estos cambios han sido positivos y la rápida evolución de estas tecnologías ha llevado a que el hombre controle diferentes dispositivos tanto en el ámbito doméstico como industrial. Internet de las cosas desde hace muchos años se la ha considerado como el paradigma tecnológico que cambiará al mundo por la capacidad de ofrecer conectividad a cualquier tipo de dispositivo hacia la internet, conformando así una gran red de máquinas que pueden interactuar entre sí; también se la denomina Internet of Everything (IoE) o Internet de todo, en español (Lee y Lee, 2015, p. 1)

Si bien en IoT, uno de los protocolos que más ha llamado la atención es MQTT, por su ligereza en cabeceras y por ende la rapidez de la transmisión en los datos. Aunque no es el único protocolo de internet de las cosas, sí que se ha ganado popularidad, luego de HTTP. Una investigación realizada en 2016 propone que en el ámbito de la transferencia de datos, HTTP ha sido ampliamente utilizado; sin embargo, su implementación en redes para el Internet de las cosas (IoT) conlleva una considerable sobrecarga. Para abordar esta problemática, se han explorado protocolos de transferencia basados en nombres. En este contexto, dicho trabajo realiza una comparativa del rendimiento entre HTTP y MQTT (Yokotani y Sasaki, 2016). Con esta investigación es posible realizar una comparativa entre un protocolo de IoT y otro protocolo de comunicación, en este caso, el protocolo HTTP.

Cabe destacar que el presente trabajo de investigación se originó a partir de un estudio interinstitucional de la corporación CEDIA en el año 2020 donde también participó la ESPOCH, que exploró la aplicación de tecnologías inmersivas y realidad virtual para enriquecer la experiencia de aprendizaje en campos como la ingeniería y la automatización industrial. La investigación de CEDIA se centró en proporcionar entornos virtuales interactivos y herramientas de apoyo, permitiendo a los estudiantes sumergirse en lecciones donde no solo pueden explorar laboratorios virtuales, sino también interactuar con otros compañeros utilizando herramientas, todo ello replicando la experiencia física en el entorno real (CEDIA, 2020). De manera que se ha pensado realizar una aplicación de realidad virtual con el mismo enfoque de enseñanza con ambientes interactivos, con la participación de un protocolo IoT para toda la comunicación.

Asimismo, existe una investigación del año 2019 donde se introduce el término VRITESS (VR-IoT Environment Synchronization Scheme) que, traduciendo al español, es Esquema de Sincronización de Entornos VR-IoT. Pues su investigación abarca la integración de IoT en entornos 3D y la vida real, creando un ambiente en donde se sincronizan los datos manipulados tanto en la sala 3D como con el objeto real. En un contexto de VR-IoT, la sincronización del entorno puede implicar la coordinación de eventos o información entre el mundo virtual y los dispositivos del mundo real conectados a la red. Por ejemplo, se podría tener un entorno de realidad virtual que refleje información en tiempo real de sensores IoT, o viceversa. Este tipo de integración puede tener aplicaciones en diversas áreas, como la capacitación virtual, la visualización de datos en entornos virtuales, la simulación de situaciones del mundo real y más. (Simiscuka et al., 2019)

Todas las investigaciones mencionadas sientan las bases para la búsqueda de una solución potencial al acceso de servicios de realidad virtual mediante la aplicación de protocolos IoT. Este enfoque surge de la necesidad de encontrar soluciones eficaces para la transferencia de datos en entornos de Internet de las Cosas (IoT), teniendo en cuenta las restricciones inherentes al protocolo HTTP.

1.2 Limitaciones y delimitaciones

Al considerar restricciones dentro de la investigación, se busca asegurar que esta sea realista y enfoque sus esfuerzos en aspectos clave para abordar el planteamiento del problema de manera efectiva. Esto ayudará a interpretar adecuadamente los resultados obtenidos y realizar recomendaciones para futuras investigaciones y aplicaciones prácticas.

1.2.1 Limitaciones

- Recursos técnicos: una de las grandes limitaciones a la que se expone la investigación es a la de los recursos técnicos disponibles, como el hardware y el software, tanto para la implementación de la red para los casos de estudio como para la realización de las pruebas. Por lo que la escalabilidad del escenario se ve afectado directamente por la limitación de estos recursos.

- Complejidad de los entornos 3D: los escenarios creados en el motor de realidad virtual tienen cierta complejidad puesto que los elementos que se involucran son animaciones, gráficos que funcionan con física y las operaciones como la programación detrás de todo el ambiente visual, han sido un desafío. Además de esto, la integración de protocolos de comunicación

IoT sugiere un grado más de complejidad al momento de ejecutar las pruebas de rendimiento. De manera que, el escenario 3D comprenderá aspectos básicos como la utilización de un escenario que puedan intervenir 3 usuarios al mismo tiempo.

- Limitaciones del protocolo: el trabajo de investigación también comprenderá el uso del escenario para acceder a los servicios de realidad en tiempo real y también con datos almacenados previamente. Al ser HTTP un protocolo de tipo cliente-servidor, solamente al realizar una petición es posible obtener una respuesta, de manera que la aplicación de este protocolo dentro del entorno 3D en tiempo real no se podría obtener una experiencia de usuario buena ya que el usuario tendría que realizar varias peticiones de forma manual, refrescando la pantalla cada vez que necesite ver los movimientos actuales de la aplicación.

1.2.2 Delimitaciones

- Enfoque en protocolos específicos: Uno de los objetivos de investigación es el estudio de los protocolos IoT y su integración en ambientes de realidad virtual. La investigación se centrará en el diseño e implementación de dos escenarios: uno con un protocolo IoT y otro con el protocolo HTTP para el acceso a servicios de realidad virtual. El protocolo de comunicación IoT se seleccionaría adecuadamente a través de diferentes métodos y técnicas de entre todos los protocolos exclusivos de IoT.

- Ámbito de la aplicación: la investigación se enfoca exclusivamente en el acceso a servicios de realidad virtual en un entorno determinado. No se analizarán otras aplicaciones o casos de uso de IoT y realidad virtual fuera del contexto específico del acceso a servicios.

- Muestra representativa: la investigación se llevará a cabo con una muestra representativa de usuarios de servicios de realidad virtual. No se considerará un muestreo exhaustivo de todas las posibles categorías de usuarios o entornos, lo que podría limitar la generalización de resultados a una población más amplia. Se ha establecido un límite en cuanto al número de usuarios que intervienen dentro del juego de realidad virtual, esto se ha definido de acuerdo con el enfoque que le han dado los tutores y el investigador: elaborar el escenario de red tal que, pueda ser orientado al uso dentro de la institución a modo de una aplicación interactiva entre 2 estudiantes y un profesor.

- Escenario de prueba específico: El escenario de prueba se diseñará para simular situaciones específicas de acceso a servicios de realidad virtual, y no abordarán todas las variaciones posibles en el uso de la tecnología IoT y realidad virtual. Al definir los actores que intervienen

en el escenario y el objetivo del trabajo, fue necesario la utilización de dos aplicaciones: una de ellas para dos estudiantes (o agentes que pueden generar la información), y la otra aplicación destinada al uso de un profesor (o agente que pueda observar los movimientos generados en tiempo real y los almacenados por el estudiante). El entorno 3D de la aplicación de los estudiantes tiene las funciones de subida de datos hacia una base de datos para almacenarlos, esto es, el estudiante tendrá 3 intentos para enviar los movimientos y cada intento puede enviar 5 movimientos obligatorios; y al mismo tiempo debe enviar datos de los movimientos en tiempo real hacia la aplicación del profesor. En tanto, la aplicación del profesor tiene las funciones de acceder a los servicios de realidad que sería la de ver los movimientos generados por los estudiantes tanto en tiempo real como los almacenados por los estudiantes.

- Dispositivos IoT: Un aspecto que se ha dejado fuera de este trabajo es el uso de dispositivos exclusivo de IoT como sensores, actuadores y demás. Puesto que el propósito es la integración de un protocolo IoT en aplicaciones de realidad virtual, las aplicaciones que pueden estar en un teléfono celular como en un computador y estos dispositivos tendrían un identificador que sería reconocido como un cliente IoT.

- Heterogeneidad de protocolos: En lo posible, se evitaría usar una combinación de protocolos IoT para evaluar resultados de manera aislada. Esto es, utilizar el protocolo IoT seleccionado en todas las funciones que se lleguen a crear en la aplicación 3D y escenario de red.

- La visualización del entorno en tiempo real: Ambas aplicaciones utilizarán el mismo entorno 3d, pero con diferentes funciones en su programación para cada protocolo.

- Servidores e intermediarios: Para que pueda existir la comunicación entre aplicaciones, debe existir un ente que coordine la comunicación de los datos. Este ente puede ser un servidor o intermediario al cual se conectan los usuarios para compartir información y debe ser un software Open Source gratuito. Además, el servidor o intermediario debe estar alojado en la nube por motivos de escalabilidad a largo plazo.

- Seguridades: el presente trabajo no tiene mecanismo de seguridad alguno como autenticación o encriptación, el caso de estudio se basa principalmente en la comparación de resultados entre dos protocolos en su comunicación más básica. Esto es porque al no considerar la seguridad se puede evaluar la capacidad de ambos protocolos para manejar diferentes cargas de tráfico y poder identificar posibles limitaciones o cuellos de botella en términos de latencia y rendimiento.

1.3 Problema General de Investigación

¿El acceso a servicios de realidad virtual será más eficiente al implementar una red orientada a IoT?

1.4 Problemas específicos de investigación

¿Cuáles son los protocolos de comunicación más usados en entornos IoT?
¿Qué problemas surgen al utilizar distintos protocolos de comunicación en las plataformas IoT?
¿Cuáles son las ventajas de utilizar protocolos IoT en redes de acceso a servicios?
¿Cómo aporta la implementación de un servidor bróker en redes de acceso a servicios?

1.5 Objetivos

1.5.1 Objetivo general

Diseñar e implementar una red para acceso a servicios de realidad virtual orientado al IoT

1.5.2 Objetivos específicos

- Revisar la literatura de protocolos e infraestructura IoT.
- Analizar los protocolos IoT y compararlos frente a otros protocolos de comunicación.
- Diseñar la red para acceso a servicio de realidad virtual orientado a IoT.
- Implementar la red para acceso a servicios de realidad virtual de acuerdo a los requerimientos del dimensionamiento de la red.
- Evaluar el desempeño del sistema para acceder a servicios de realidad virtual

1.6 Justificación

1.6.1 Justificación teórica

La justificación teórica de esta investigación radica en la relevancia que tienen tanto la realidad virtual como la Internet de las cosas (IoT) en la actualidad. La realidad virtual ha emergido como una tecnología disruptiva y prometedora que ha transformado la manera en que interactuamos con el mundo digital. Por otro lado, el Internet de las cosas ha revolucionado la forma en que los dispositivos y objetos cotidianos se conectan e intercambian datos, permitiendo la automatización

y optimización de diversos procesos. La convergencia de estas dos tecnologías ofrece una oportunidad única para mejorar la experiencia del usuario en aplicaciones de realidad virtual. (Simiscuka et al., 2019; Lee y Lee, 2015)

En la medida en que la realidad virtual se expande hacia sectores como la educación y enseñanza, el entretenimiento, la medicina y la industria; la eficiencia y la calidad del acceso a servicios de realidad virtual se convierten en aspectos críticos. La adopción de una red orientada al IoT para el acceso a estos servicios podría proporcionar ventajas significativas, como una mayor velocidad de transmisión de datos, menor latencia y una mejor gestión del ancho de banda (Radianti et al., 2020). La justificación teórica de esta investigación se fundamenta en la premisa de que la implementación de un protocolo de comunicación IoT podría optimizar la entrega de datos en tiempo real, lo que resultaría en una experiencia de realidad virtual más inmersiva y fluida para los usuarios.(Yokotani y Sasaki, 2016)

En la mayoría de las aplicaciones software se utiliza el protocolo HTTP, pues es el mayormente difundido en el desarrollo de distintas aplicaciones para crear enlaces de comunicación entre un servidor y un cliente. Desde buscar en un navegador hasta abrir una aplicación en el teléfono móvil usan HTTP en su programación interna para establecer distintas conexiones con un servidor. El análisis comparativo de protocolos de comunicación IoT con el protocolo HTTP tradicional aportará conocimientos valiosos para la comunidad de investigación y desarrollo (Yokotani y Sasaki, 2016). Esto permitirá una comprensión más profunda de las implicaciones y ventajas de utilizar protocolos de IoT en redes de acceso a servicios de realidad virtual. La investigación también contribuirá a identificar las limitaciones y posibles desafíos que podrían surgir al implementar soluciones de IoT en este contexto, lo que puede orientar futuras investigaciones y mejoras en el diseño de redes. (Yokotani y Sasaki, 2016; Behal et al., 2023; Simiscuka et al., 2019)

Existe evidencia que este tipo de convergencia entre aplicaciones en 3D e IoT se lo ha utilizado para compartir información entre Unity 3D con un elemento físico como la ESP32, Arduino, Node MCU, entre otras; mediante librerías de protocolos IoT (Simiscuka et al., 2019). Con esto expandimos el conocimiento y existe una premisa de que sí es factible realizar este tipo de combinación por medio de librerías de programación, mas no se tiene datos de su rendimiento y eficiencia. Se espera que los hallazgos de esta investigación contribuyan al avance de la tecnología de realidad virtual y aporten a la comunidad científica y tecnológica nuevos conocimientos que respalden la implementación de soluciones innovadoras y eficientes en este campo interdisciplinario.

1.6.2 Justificación Metodológica

Para llevar a cabo el desarrollo de esta investigación se han elegido diferentes metodologías que son necesarias para que los resultados sean válidos y confiables. El investigador ha estructurado y guiado el proceso de investigación aplicando el método científico. Para ello, ha formulado una pregunta de investigación clara y ha realizado una exhaustiva revisión de la literatura relevante relacionada con la integración de protocolos de comunicación IoT en entornos 3D en Unity. Con base en el conocimiento adquirido, el investigador ha formulado hipótesis o suposiciones iniciales que serán puestas a prueba en el desarrollo de la investigación.(Hernández-Sampieri y Mendoza, 2018; Simiscuka et al., 2019)

La investigación documental ha sido fundamental para obtener información detallada sobre los protocolos de comunicación IoT y otros temas relevantes para el proyecto. El investigador ha revisado libros electrónicos, fuentes en internet y otros trabajos concluidos que han sido de vital importancia para orientar el desarrollo de la investigación y mejorar la calidad del escrito.

Asimismo, se ha aplicado el método analítico para descomponer el problema de investigación en componentes más pequeños y analizarlos individualmente (Hernández-Sampieri y Mendoza, 2018). Se han examinado diversos aspectos de la integración de protocolos de comunicación IoT en entornos 3D, como la compatibilidad, el rendimiento y la latencia. Esto ha permitido al investigador obtener una comprensión profunda de cada aspecto y su influencia en la comunicación efectiva entre dispositivos IoT y entornos 3D en Unity.(Simiscuka et al., 2019)

En cuanto al método experimental, se han diseñado casos de estudio específicos y se han implementado prototipos en Unity que integran los protocolos de comunicación IoT seleccionados. El investigador ha recopilado datos en tiempo real para evaluar el rendimiento, la latencia y la escalabilidad de la solución propuesta. Durante este proceso, se han aplicado técnicas de control de variables y se han realizado comparaciones cuantitativas entre diferentes configuraciones y condiciones.

Además, se ha empleado el método inductivo para recopilar datos y observaciones específicas en los casos de estudio (Hernández-Sampieri y Mendoza, 2018). A partir de estos datos, se han identificado patrones, tendencias y generalizaciones relevantes para el estudio en cuestión. De igual manera, el método deductivo ha sido utilizado para establecer premisas o principios generales basados en teorías previas relacionadas con la integración de protocolos de comunicación IoT.(Hernández-Sampieri y Mendoza, 2018; Behal et al., 2023)

1.6.3 Justificación Práctica

En el presente trabajo de investigación como objetivo principal se establece diseñar una red para acceso a servicios de realidad virtual orientado al IoT, lo que se pretende es permitir el acceso a una aplicación de VR con el uso de algún protocolo de comunicación IoT que facilite la transmisión de mensajes en tiempo real y sin pérdida de datos en el mejor de los casos. (Simiscuka et al., 2019)

En primer lugar, los resultados de esta investigación proporcionarán una guía práctica para diseñadores y desarrolladores de redes de acceso a servicios de realidad virtual. La identificación de las ventajas y desventajas de utilizar protocolos de comunicación IoT en comparación con el protocolo HTTP tradicional permitirá tomar decisiones fundamentadas sobre qué enfoque adoptar para optimizar el rendimiento de sus aplicaciones. Esto ayudará a mejorar la calidad de los servicios de realidad virtual ofrecidos a los usuarios, lo que a su vez podría conducir a una mayor satisfacción del cliente y una mayor retención de usuarios.

Se debe agregar que, la investigación podría ofrecer recomendaciones específicas sobre la selección de protocolos de comunicación para diferentes casos de uso de realidad virtual. Por ejemplo, se podría sugerir el uso de MQTT en aplicaciones que requieran una entrega rápida y en tiempo real de datos, como simulaciones médicas o entrenamientos virtuales en tiempo real. Por otro lado, el protocolo HTTP podría ser preferible en escenarios donde la latencia no sea un factor crítico, como en experiencias de realidad virtual más estáticas, como visitas virtuales a museos o exhibiciones. (Yokotani y Sasaki, 2016)

Los resultados de esta investigación podrían impulsar avances significativos en la integración de estas tecnologías, beneficiando tanto a la industria como a los usuarios finales, mejorando la forma en que interactuamos y aprendemos en entornos virtuales.

1.7 Hipótesis

1.7.1 Hipótesis general

¿Se considera que el acceso a servicios de realidad virtual será más eficiente al implementar una red orientada a IoT?

1.7.2 Hipótesis nula

¿El acceso a servicios de realidad virtual no será más eficiente al implementar una red orientada a IoT?

1.7.3 Hipótesis alternativa

¿El acceso a servicios de realidad virtual si será más eficiente al implementar una red orientada a IoT?

CAPÍTULO II

2. MARCO TEÓRICO

2.1 Antecedentes de investigación

En la era digital actual, la realidad virtual (RV) se ha posicionado como una tecnología innovadora y poderosa que ha revolucionado diversas industrias y sectores (Radianti et al., 2020). Desde el entretenimiento y la educación hasta la medicina y la ingeniería, la RV ha demostrado su capacidad para sumergir a los usuarios en mundos virtuales, ofreciendo experiencias inmersivas y altamente atractivas. Sin embargo, el acceso fluido y eficiente a servicios de realidad virtual sigue siendo un desafío importante para brindar una experiencia de usuario óptima y sin interrupciones. (Coogan y He, 2018; Radianti et al., 2020; Simiscuka et al., 2019)

La creciente adopción de Internet de las cosas (IoT) ha traído consigo nuevas posibilidades para mejorar la conectividad y la eficiencia en diferentes entornos (Lee y Lee, 2015). La comunicación entre dispositivos suele ser rápida y fiable gracias a que estos protocolos de comunicación IoT fueron diseñados para transmitir datos ligeros en ambientes con bajo ancho de banda (González García, 2017). La integración de IoT con la realidad virtual podría ofrecer un potencial considerable para optimizar el acceso a servicios de RV y proporcionar interacciones más realistas y personalizadas para los usuarios (Simiscuka et al., 2019). En este contexto, surge la cuestión central de si implementando una red orientada al IoT mejoraría la eficiencia del acceso a servicios de realidad virtual. (Yokotani y Sasaki, 2016; Behal et al., 2023)

Para abordar este problema, es esencial examinar a fondo los protocolos de comunicación utilizados en entornos IoT y comprender sus implicaciones y ventajas. Los protocolos IoT, como MQTT (Message Queuing Telemetry Transport) y CoAP (Constrained Application Protocol), se han destacado por su capacidad para soportar redes de dispositivos interconectados con baja potencia y ancho de banda limitado, aspectos clave en la implementación de IoT (Behal et al., 2023). Estos protocolos presentan características específicas que pueden influir en la eficiencia y confiabilidad en el contexto de acceso a servicios de realidad virtual. (González García, 2017; Yokotani y Sasaki, 2016)

Por otro lado, el protocolo HTTP ha sido ampliamente utilizado en el acceso a servicios web y tradicionalmente ha sido la opción preferida para la comunicación entre clientes y servidores. Sin embargo, su naturaleza de solicitud-respuesta podría presentar limitaciones en la entrega oportuna y continua de datos en entornos de realidad virtual, lo que puede afectar negativamente la experiencia del usuario.(Gemirter et al., 2021)

Mediante una investigación profunda y sistemática, se busca comparar el rendimiento de los dos escenarios implementados: uno basado en un protocolo IoT seleccionado en base a sus características y otro en el protocolo HTTP (Gemirter et al., 2021).

2.2 Referencias Teóricas

2.2.1 Redes de computadoras

2.2.1.1 Introducción a las redes de computadoras

Las redes de computadoras es una parte fundamental para cualquier estudio relacionado con la informática y las telecomunicaciones. La evolución de las redes de computadoras ha sido significativa desde sus inicios. Las primeras redes surgieron en la década de 1960, como el ARPANET, un proyecto financiado por el Departamento de Defensa de los Estados Unidos. De tal manera que las redes de computadoras son sistemas que permiten la comunicación y el intercambio de información entre diferentes dispositivos, ya sea a nivel local o global, pues, estas redes son esenciales para el desarrollo de la informática y la conectividad, permitiendo la transferencia de datos, el acceso a recursos compartidos y la comunicación en tiempo real. (Sánchez Rubio et al., 2020)

Las redes de computadoras han evolucionado a tal punto que en la sociedad moderna son un elemento clave o fundamental para la vida cotidiana de todas las personas como el trabajo, entretenimiento, la educación y la investigación. (Forouzan, 2012; Patterson y Hennessy, 2011)

2.2.1.2 Definición

Al ser un tema amplio existen diversas definiciones y conceptos. Según la Universidad del Sur de Florida, las redes "Una red consiste en dos o más computadoras que están conectadas para compartir recursos […] intercambiar archivos o permitir comunicaciones electrónicas.[1]" (University of South Florida, 2013)

En otras palabras, las redes de computadoras son sistemas de comunicación que permiten la interconexión de dispositivos electrónicos como computadoras, impresoras, routers, switches y servidores, entre otros. Estas redes se utilizan para compartir recursos como archivos,

[1] Traducido del inglés de la página web de la Universidad del Sur de Florida

aplicaciones y servicios, y para la comunicación entre usuarios y sistemas. Además, permite el acceso a Internet y a otros sistemas remotos. (Forouzan, 2012)

2.2.1.3 Comunicación de datos

Dentro de las redes, se menciona varias veces la comunicación o intercambio de los datos a través de diversos medios. La comunicación de datos básicamente es un proceso donde se intercambia información entre dos dispositivos por un medio de transmisión que puede ser guiado o no guiado o en su defecto, cableado o inalámbrico; utilizando protocolos de comunicación y estándares que garantizan la interoperabilidad entre diferentes dispositivos y sistemas. (Stallings, 2014)

El cómo se realiza y qué componentes tiene es la clave principal para entender a fondo todo lo relacionado a las redes y su funcionamiento con más claridad. No porque sea un proceso fácil de percibir quiere decir que no pueda ser complejo de aplicar, pues se necesita de otros factores para que se pueda transmitir la información. (Tanenbaum y Wetherall, 2012)

- Propiedades de la comunicación de datos

La comunicación de datos es un aspecto clave en las redes de computadoras, que requiere una transmisión efectiva y eficiente de la información. En este sentido, es esencial que la comunicación cumpla con ciertas características, tales como la entrega, precisión puntualidad y orden de los datos. (Forouzan, 2012)

La entrega de los datos debe estar dirigidos hacia determinado objetivo, para lo cual es fundamental que la entrega del mensaje sea al destinatario o destinatarios correctos. O a su vez, estos no se deben perder en el camino y el destinatario nunca reciba el mensaje. (Tanenbaum y Wetherall, 2012)

La precisión consiste en entregar un mensaje con datos íntegros o información sin errores, es decir, no introducir datos erróneos que pueda interferir en el entendimiento del mensaje (Stallings, 2014). Un ejemplo análogo a esto último seria que, en una carta escrita a mano, el mensajero que la lleva abra la carta y modifique parte de su texto entregando un mensaje erróneo o corrompido a su destino. Esto es exactamente lo contrario a precisión y lo que se debe evitar en la comunicación de datos. Esta propiedad es denominada de otra manera a la obra Data and Computer Communication como integridad de los datos pues consiste en la misma ideología, enviar información integra y llegue sin corrupción alguna. (Stallings, 2014)

La puntualidad es un detalle muy importante para considerar, debido a que se deben realizar en el menor tiempo posible, más aún si se trata de enviar mensajes en tiempo real, el retardo debe ser el mínimo. Esto va de la mano con el concepto de QoS o calidad del servicio, en donde se priorizan los datos de ciertos servicios para que se envíen en el menor tiempo posible como los datos de video streaming. (Stallings, 2014; Sánchez Rubio et al., 2020)

Lo que se puede decir en el aspecto de orden, es que si se entregan varios mensajes con información consecutiva no se deben entregar de forma desordenada pues la información transmitida seria imprecisa o incongruente. De manera que el orden de entrega de la información debe seguir el mismo patrón desde que sale el mensaje hasta que llega al destinatario. (Dordal, 2020; Stallings, 2014)

Dadas todas estas propiedades, se puede concluir que lo que se busca en una red de computadores es que la información que se envía a un destinatario debe ser eficiente y fiable.

2.2.1.4 Medios de transmisión

O también llamado canales de comunicación, es el lugar por donde circula o se transmite información a manera de mensajes. Desde el punto de vista técnico se dividen en dos grandes grupos: medio de transmisión cableado o guiado, y el medio de transmisión inalámbrico o no guiado (Telefónica I+D, 2002). A su vez puede estar conformada por una sola línea, o por varias líneas como una red compleja que en conjunto pueden ser homogéneas (de un solo tipo) o heterogéneas (de diferentes tipos). (Stallings, 2014)

Los medios guiados o cableados son todos los canales físicos como el par de cobre y la fibra óptica. Este medio puede ser una línea dedicada a otorgar conexión privada, o puede ser que esta lleve información multiplexada de varios usuarios en un solo canal. (Forouzan, 2012)

El acceso no guiado por su parte no utiliza ningún medio físico, utiliza el aire para transportar los datos en forma de ondas electromagnéticas. (Telefónica I+D, 2002)

a. Medios de transmisión guiados

En este apartado se pueden distinguir los medios guiados por par de cobre y la fibra óptica. Ambos son medios guiados pero el de material de cobre envía señales eléctricas, mientras que la fibra óptica funciona por medio de haces de luz. (da Silva, 2016)

- **Par de cobre:**

La tecnología de par de cobre reinaba en las redes de comunicaciones hace más de 2 décadas. Fue la tecnología con la que apareció los primeros indicios de comunicación como la telefonía para transportar voz en formato analógico (da Silva, 2016). (Ver Ilustración 2-I)

Ilustración 1-2: Cable par de cobre con chaqueta

Realizado por: Huilcamaygua, Cinthya, 2023.

Con el pasar de los años los requerimientos de los usuarios fueron aumentando y con ello se desarrollaron técnicas para solventar esos requerimientos, sin duplicar la inversión (Escalona, 2012), hasta llegar a las conocidas xDSL. A continuación, se mencionarán algunas de ellas:

Acceso telefónico convencional: Ideal para el servicio telefónico e internet a través de módem (Figueiras Vidal, 2002), esta tecnología se encuentra casi obsoleta porque aún se sigue usando en algunas zonas de países en vías de desarrollo. (Ver Ilustración 2-2)

Ilustración 2-2: Diagrama de tecnologías DSL.

Fuente: (ARCOTEL, 2019b, p. 10)

ADSL es una de las tecnologías/técnicas ampliamente difundidas, funciona a distancias entre 2 a 5.5 Km, su ancho de banda asimétrico bordea el 1 MHz ofreciendo una tasa de subida de 1.5 y 6. Mbps; y una tasa de bajada entre los 16 a 140 Kbps. (Telefónica I+D, 2002)

VDSL es lo último en técnicas del par de cobre, es simétrico, mejoró por mucho la tasa de transferencia en un promedio de 26 Mbps de subida y de bajada. Está diseñada para distancias

cortas con un máximo de 300 metros. Se permitió la transmisión servicios multimedia, voz y datos.(Figueiras Vidal, 2002; Telefónica I+D, 2002)

- **Cable coaxial:**

Este medio de transporte de información está formado por 2 conductores: el conductor interno se encuentra dentro de un material aislante, que a use vez este aislante está contenido por una capa concéntrica del otro conductor en forma de malla. A este último conductor lo cubre una chaqueta, que no es más que un material aislante que lo va a proteger de agentes externos que podrían ser señales eléctricas externas o interferencias (Sánchez Rubio et al., 2020; Forouzan, 2012). (Ver Ilustración 3-2)

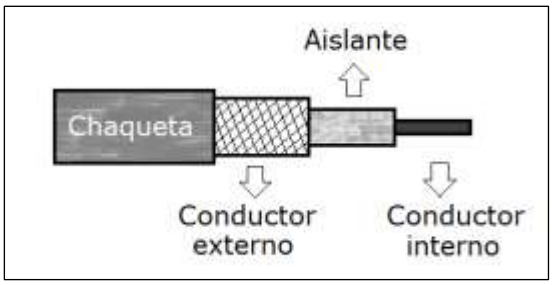


Ilustración 3-2: Estructura del cable coaxial.

Fuente: (Sánchez Rubio et al., 2020, p.52)

Realizado por: Huilcamaygua, Cinthya, 2023.

Al ser un cable con capas concéntricas de conductor-aislante-conductor, su impedancia está determinada por la distancia entre los dos conductores y el tipo de material aislante, proporcionando características al cable para transmitir información a altas velocidades, superando las velocidades del par de cobre. (da Silva, 2016; Sánchez Rubio et al., 2020, p. 52)

- **PLC (Power Line Communications):**

Utiliza como transporte físico las líneas eléctricas de baja tensión, ya que al ser la infraestructura eléctrica más grande a nivel mundial no se requiere de equipamiento adicional (Ver Ilustración 4-2), el principal requisito es que sea de baja tensión (Escalona, 2012).

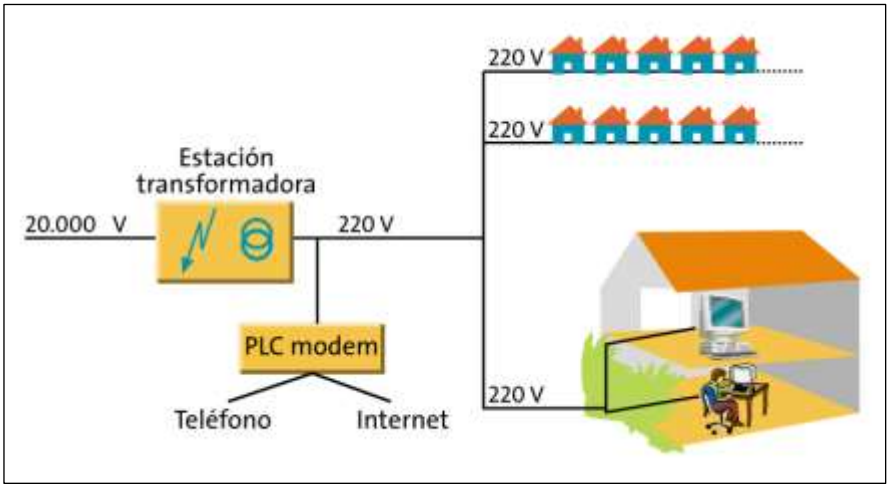


Ilustración 4-2: Acceso a teléfono e internet por la red eléctrica de baja tensión.

Fuente: Telefónica I+D, 2002, p.237.

Puede llegar a tener una velocidad de transmisión desde 2 hasta 45 Mbps. Utilizan bandas de frecuencias altas de 2 a 30 MHz para evitar que los datos interfieran con el servicio eléctrico. La principal limitación del uso de esta tecnología es que no toda la red eléctrica está diseñada para trasmitir datos. (Escalona, 2012)

Sin embargo, sigue siendo una excelente opción porque se puede la aprovechar para servicios de banda ancha con multiplexación OFDM y la técnica Mi-Mo (Hu et al., 2014). Las investigaciones más recientes acerca de los sistemas PLC quiere tener una convergencia con las redes Wi-Fi y arrojan resultados favorables (De Beer et al., 2016). Mientras que para el mercado automovilístico ven conveniente usar esta técnica para las baterías inteligentes sin el uso de cables adicionales.(Landinger et al., 2020)

- **Cable modem o HFC (Hybrid Fiber Coaxial):**

Fue dirigido para el servicio de televisión por cable o CaTV[2] como sustituto de la televisión analógica terrestre. Utiliza fibra óptica y cable coaxial, se ve limitado principalmente por la distancia donde tienen que colocar amplificadores para el coaxial cada cierta longitud hasta llegar al abonado (Ver Ilustración 5-2). En la actualidad funciona como medio de trasmisión para brindar el servicio de Internet fijo, aunque también puede ser multiservicio y puede brindar las mismas tasas de transferencia que la tecnología ADSL si tiene un adecuado dimensionamiento.(González, 2012)

[2] Redes de televisión por cable

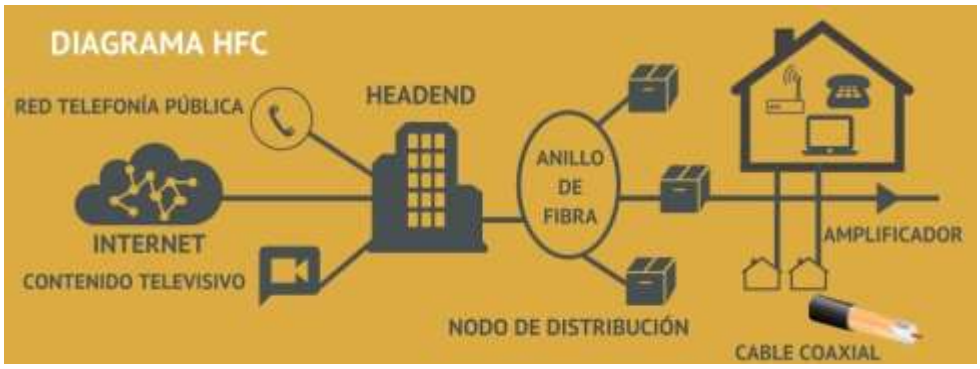


Ilustración 5-2: Diagrama de la red hibrida de coaxial y fibra óptica.

Fuente: ARCOTEL, 2019, p.10.

- **Fibra óptica:**

Es la solución actual de las redes de comunicaciones por su capacidad de transportar tráfico, extendido tiempo de vida y menor costo del material. Ofrece mejor calidad del servicio sin interferencias electromagnéticas porque es un elemento pasivo (ARCOTEL, 2019a).

La estructura básica de la fibra óptica se compone de 3 elementos principales: Chaqueta o jacket, revestimiento o cladding y núcleo o core. (Ver Ilustración 6-2). Aunque esta composición puede variar dependiendo del número de cables y el uso al que se destine.

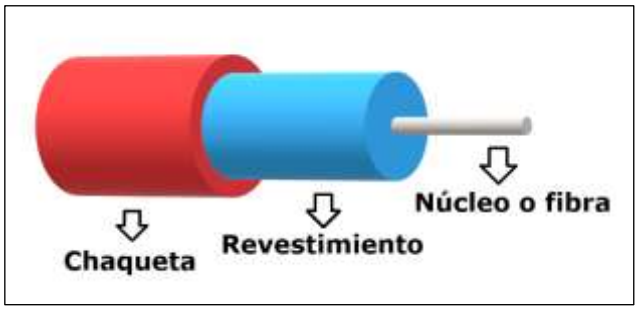


Ilustración 6-2: Estructura básica de la fibra óptica.

Fuente: (Sánchez Rubio et al., 2020, p. 53)

Realizado por: Huilcamaygua, Cinthya, 2023.

Además de estas ventajas, se pueden mencionar otras como el tamaño del diámetro del cable y su peso, pues es delgado y ligero; también que se puede fabricar de materiales como el plástico y el vidrio que se pueden obtener fácilmente y por último se puede indicar que en este medio no existe diafonía.(da Silva, 2016)

Se la puede dividir por clases como lo son FTTH (Fiber To The Home) que es la fibra que llega hasta el hogar del usuario; FTTB (Fiber To The Building) que es la fibra que llega hasta la

infraestructura de un edificio; FTTC (Fiber To The Cabinet) que es la fibra que llega hasta la acera; todas estas se pueden resumir en FTTx ya que todas se refieren la llegada del servicio lo más cercano posible al usuario, como se puede ver en la Ilustración 7-2. (Telefónica I+D, 2002)

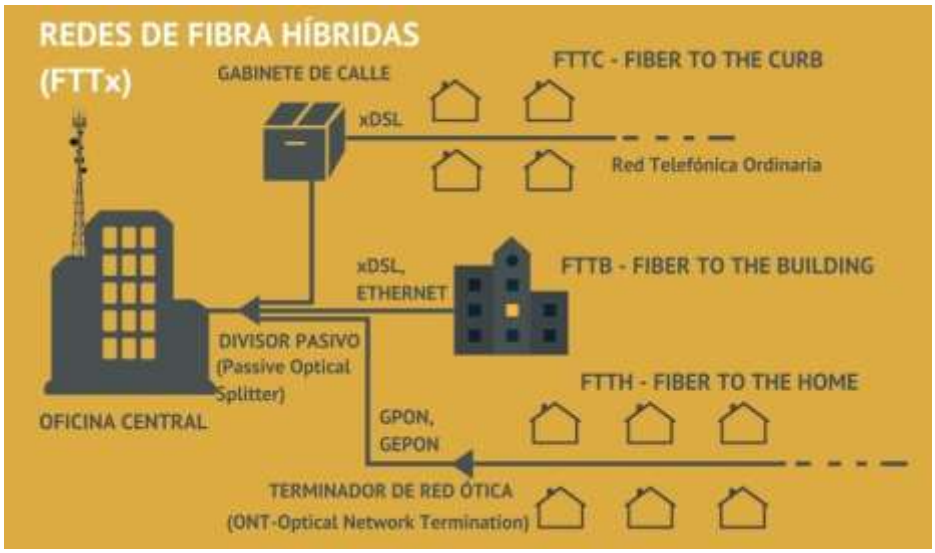


Ilustración 7-2: Diagrama de la red hibrida FTTx.

Fuente: ARCOTEL, 2019, p.10.

La gran velocidad de transmisión que maneja y el abaratamiento de esta tecnología ha logrado que sea la más aclamada por todos, disfrutando así de todos los servicios sean multimedia, voz, datos, mensajería y juegos en línea. (ARCOTEL, 2019)

Actualmente hay estudios donde se pretende extender el alcance de la transmisión que puede tener la fibra óptica mediante POF[3] y de acuerdo con el estudio, si se ha logrado su cometido, con el fin de abaratar los costos de los amplificadores de la señal óptica. (Ali et al., 2019)

b. Medio de transmisión no guiado o inalámbrico

La tecnología inalámbrica no usa un medio físico para la transmisión. El medio utilizado es vía aire. Son redes de radiofrecuencias de diferentes valores de longitud de onda para transmitir la información. La movilidad del usuario requiere una conexión inalámbrica. Por otra parte, las redes inalámbricas suelen ser más susceptibles a interferencias electromagnéticas producidos por sistemas eléctricos y condiciones climáticas como la radiación solar lluvia y tormentas eléctricas (Telefónica I+D, 2002). Las redes Wi-Fi es un gran ejemplo de medio de transmisión no guiado, así

[3] Plastic Optical Fiber o fibra óptica de plástico

como WiMAX, pues las ondas electromagnéticas se transportan a través del aire. Al igual que lo hace la tecnología Bluetooth, ZigBee, LoRa, entre otras. (Córdova, 2008)

Las comunicaciones inalámbricas se pueden realizar de 3 maneras: por medio de ondas de radio, por microondas y ondas infrarrojas. (da Silva, 2016)

- **Ondas de Radio:**

Las ondas de radio fueron el primer medio de transmisión no guiada utilizada. Son ondas electromagnéticas que van desde la frecuencia de los 3 kHz hasta los 1 GHz, generalmente son ondas omnidireccionales de acuerdo con el rango de frecuencias que utiliza pues se distribuyen por todas direcciones, de manera que para enviar y recibir información no es necesario que ambos equipos se encuentren con línea de vista o estén alineados para compartir la información (da Silva, 2018). Algo muy destacable de este tipo de ondas es que podría recorres grandes distancias y atraviesa las paredes ya que las bajas frecuencias con las que trabaja pueden pasar por casi cualquier obstáculo, inclusive en el caso hipotético de utilizar una antena a gran potencia, la señal puede dar la vuelta al mundo sin problemas. Dentro de esta categoría se hallan las señales de radio AM, FM, de TV análoga, de celular, de dispositivos de walk and talk, Wi-FI, entre otras. (Regalado Jalca et al., 2018, p. 22)

- **Señales microondas:**

Las señales de microondas son señales electromagnéticas que se hallan en un rango de frecuencias de 1 GHz y 300 GHz. Este tipo de ondas es de tipo direccional por su haz de radiación muy estrecho, de modo que los dispositivos que quieren compartir información deben estar alineados y con línea de vista directa. Esta señal no puede atravesar obstáculos, ni paredes ni montañas, por esa razón necesita de línea de vista directa. Las señales de las redes móviles entre antenas del mismo tipo están dentro de esta categoría. (da Silva, 2016; Regalado Jalca et al., 2018, p. 23)

- **Señales Infrarrojas:**

Las señales infrarrojas son señales de muy corto alcance, cuya frecuencia se halla entre los 300 GHz y 400 GHz. Se conoce muy bien que altas frecuencias no pueden atravesar obstáculos, por lo que no es difícil intuir que estas señales no pueden penetrar aquellos obstáculos. Se caracterizan por tener muy buena tasa de transferencia y es apta para ser utilizada en dispositivos como periféricos de las computadoras (ratón, teclados). (da Silva, 2016)

2.2.1.5 Tipos de redes

En la actualidad, existen diferentes tipos de redes de computadoras, como las redes de área local o LAN, las redes de área amplia o WAN, redes inalámbricas como el wifi, entre otras. Todas ellas han sido clasificadas según el área de cobertura o alcance que tiene con respecto a la distancia física entre el dispositivo final con su nodo. Que a su vez pueden ser cableadas o no cableadas, es decir, puede tener un medio guiado cableado físico o un medio no guiado inalámbrico, tal como se muestra en la Ilustración 8.2

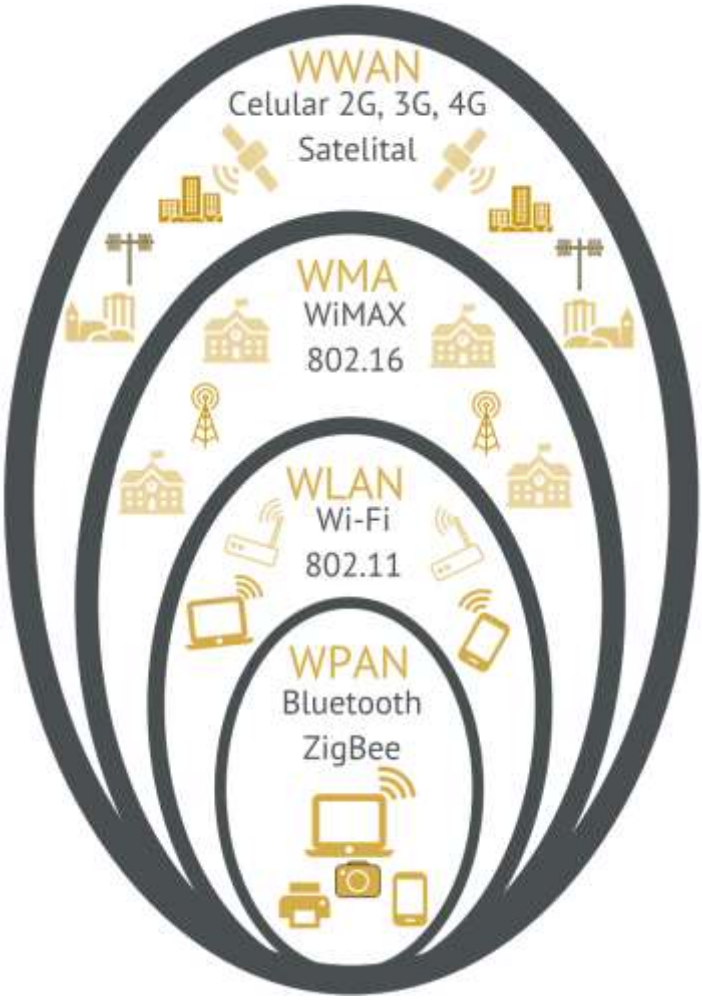


Ilustración 8-2: Tecnologías Inalámbricas por área de cobertura.

Fuente: ARCOTEL, 2019, p.13.

- **LAN**

Las redes LAN (Local Area Network) o redes de área local tiene un alcance medio entre 10 y 100 metros, prácticamente cubre una pequeña área como lo puede ser el piso de una vivienda o el área de trabajo en una oficina, o hasta un pequeño edificio. Por otro lado, este tipo de redes también

son llamados coloquialmente como de área doméstica por el alcance que tiene. (Sánchez Rubio et al., 2020)

Las LAN tienen 2 configuraciones, pueden ser LANs conmutadas o LANs inalámbricas. En el caso de las LAN conmutadas, la tecnología que lidera es Ethernet, pero con la llegada de la comunicación inalámbrica se convirtieron en WLAN (Wireless Local Area Network) regido por el estándar IEEE 802.11, reemplazando en gran medida la tecnología Ethernet por Wi-Fi y que comúnmente se la puede encontrar en cualquier sitio. (Stallings, 2014)

Todos los equipos que se conecten a la red LAN son considerados como estación de trabajo o como servidores. Mientras las estaciones de trabajo son dispositivos destinado al usuario final, los servidores son equipos mucho más sofisticados en cuanto a rendimiento y recursos computacionales que son más elevados que una estación de trabajo o PC. Hay que destacar que una sola persona u organización es dueño de la red LAN y todos los dispositivos conectados pertenecen a una misma organización, además, la gestión de una LAN generalmente recae sobre un usuario, la configuración de estas redes es relativamente sencillas por lo que los usuarios lo pueden realizar. (da Silva, 2016)

WLAN es un tipo de red que utiliza como medio de transmisión el aire a través de ondas electromagnéticas y cuya tecnología estrella es Wi-Fi estándar IEEE 802.11. se popularizo tanto que hasta hoy en día se utiliza en casi todos los hogares, oficinas, instituciones y empresas, dado que ofrece velocidades de transmisión altas y es de fácil administración. (Sánchez Rubio et al., 2020)

Las WLAN pueden subdividirse en otras redes más pequeñas como las redes PAN (Personal Area Network) y WBAN (Wireless Body Area Network). Las PAN o bien WPAN son redes que están dedicadas a la transmisión de datos en corto alcance con ayuda de la transmisión por infrarrojo. Las WBAN son redes inalámbricas un poco especiales, se aplica más en el área de la medicina, pues trata de colocar diversos sensores alrededor del cuerpo de una persona y que estos sensores se conecten de forma inalámbrica a un equipo que puede ser un computador al que le llega la información o mediciones del cuerpo. (Tanenbaum y Wetherall, 2012)

- MAN

Metropolitan Area Network o Red de área metropolitana, este tipo de red es la que ayuda a unir o conectar 2 o más redes LAN. Prácticamente ayuda de puente para la comunicación entre las LANs y las WANs. Permite la masificación de usuarios, en otras palabras, abarca miles de usuarios. Así como las redes LAN, también puede transmitir la información por un medio no

guiado o inalámbrico denominándose WMAN con el estándar IEEE 802.16 reemplazando las tecnologías cableadas ADSL, enlaces T1/E1, cable modem, entre otros. El área geográfica que cubre es grande, llegando a abarca vecindarios, ciudades y áreas metropolitanas. (da Silva, 2018, p. 122; Tanenbaum y Wetherall, 2012)

Algo importante de las redes MAN es que resulta muy caro de desplegar y mantener, después de todo, los que lo hacen generalmente son proveedores de servicios privados (da Silva, 2016). A la red WMAN la representa la tecnología WiMAX o estándar IEEE 802.16. WiMAX se diferencia mucho del concepto de Wi-Fi en cuanto al alcance, más bien se asemeja a una red de telefonía celular. El autor Leon Couch considera que WiMAX es WMAN pues es la única tecnología que lo representa. WiMAX. (Couch, 2013)

- WAN

Redes WAN o redes de área Amplia, como su propio nombre lo dice, abarca una gran extensión en cuanto a distancia física llegando a conectar áreas geográficamente grandes. Estas redes se caracterizan por cubrir áreas del tamaño de una ciudad o un estado, puesto que incorporan a las redes MAN y LAN. (Fitzgerald et al., 2011; Stallings, 2014)

Consta de varios nodos de comunicación interconectados unos con otros y es el medio por donde circula toda la información generada desde un equipo hasta otro ubicado en cualquier lugar del planeta. La infraestructura física está conformada por el cableado transoceánico o también por las señales de satélite aptas para conectar esta la red con otras a nivel global. De esta manera dos usuarios que viven en partes del mundo diferentes pueden conectarse a esta red y comunicarse en tiempo real.(da Silva, 2016)

La conformación de este tipo de redes es muy complicada su despliegue conlleva a costos muy grandes, pues une continentes enteros y en cada continente se unen varias redes en donde intervienen equipos enrutadores, multiplexores y puentes, así como las diferentes técnicas de multiplexación, conteniendo grandes cantidades de información de las redes MAN y LAN que pasan por el cableado submarino facilitando la trasportación de datos. (Tanenbaum y Wetherall, 2012, p. 20)

Para implementar las redes WAN, se usaban 2 tecnologías: conmutación de paquetes y conmutación de circuitos. Luego las tecnologías que mejor se adaptaban para este tipo de redes fueron Frame Relay y ATM. Ahora, se están desplazado de a poco hacia servicios basados en Gibabit Ethernet. (Stallings, 2014)

Un ejemplo de redes WWAN son las redes de telefonía celular, puesto que comunican a dos individuos de dos partes del mundo. Las redes de telefonía móvil son las más conocidas para conseguir conectividad de datos y otros servicios. En la actualidad la red que se ha utilizado es la LTE en Ecuador. Las tecnologías de radio móviles también son un ejemplo de redes WWAN y han ido en constante evolución para incrementar los anchos de banda disponibles mejorando cada vez la calidad y proporcionando servicios adicionales.(ARCOTEL, 2019a)

2.2.1.6 Topologías de red

Una topología de red se refiere a la estructura física o lógica de los dispositivos que conforman una red, determinando la forma en que los mismos se conectan y comunican entre sí. Se puede considerar como un plano o un mapa que define la relación entre los nodos de una red, incluyendo servidores, ordenadores y otros dispositivos de red. Las diferentes topologías de red tienen diferentes ventajas y desventajas, de manera que la elección de una topología depende de las necesidades y requerimientos específicos de la red. (Kurose y Ross, 2017; Tanenbaum y Wetherall, 2012)

La topología de red también puede ser física o lógica. La topología física se refiere a como están conectados físicamente los dispositivos, mientras que la topología lógica se refiere a como se comunican los dispositivos en la red independientemente de su ubicación física. Estos pueden ser en estrella, en bus, en anillo, en malla, etc. (Kurose y Ross, 2017)

La selección de la topología es un aspecto critico en la planificación, diseño e implementación de una red, y puede afectar la eficiencia, fiabilidad y seguridad de la red en conjunto. (Forouzan, 2012)

- **Topología de red en bus**

También llamada lineal, es de las más simples y económicas utilizadas en la interconexión de dispositivos de red. En esta topología, todos los dispositivos se conectan a una única línea de comunicación, llamada bus, que transmite los datos entre ellos. Cada dispositivo está conectado al bus a través de un conector, que es responsable de transmitir los datos a través de la línea de comunicación. Aunque esta topología es fácil de implementar, su rendimiento disminuye cuando se agregan muchos dispositivos a la red. Ya que el ancho de banda se debe compartir entre todos los dispositivos conectados. (Tanenbaum y Wetherall, 2012; Forouzan, 2012)

En la topología en bus, el fallo de un dispositivo puede afectar a toda la red ya que, si un dispositivo falla o su conector se desconecta, la línea de comunicación se rompe y todos los

dispositivos en la red pierden la conectividad. A pesar de sus limitaciones, la topología en bus sigue siendo comúnmente utilizada en pequeñas redes locales debido a su bajo costo y simplicidad de instalación, pues todos los dispositivos se conectan por medio de un cable (Sánchez Rubio et al., 2020, p. 98), tal como se observa en la Ilustración 9-2.

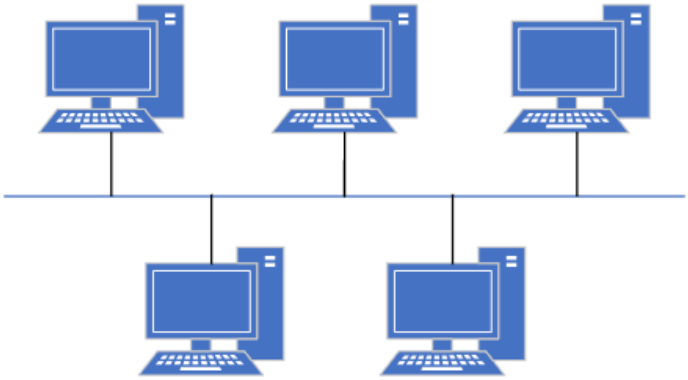


Ilustración 9-2: Topología de red en bus.

Fuente: (Sánchez Rubio et al., 2020, p. 98)

Elaborado por: Huilcamaygua, Cinthya, 2023.

- **Topología de red en anillo**

La topología en anillo es de las topologías más antiguas y fue utilizada en las primeras redes de área local (LAN). En este tipo de diseño de red cada dispositivo está conectado directamente con el dispositivo anterior y el siguiente, formando un anillo lo que crea una red cerrada (Ver Ilustración 10-2), de manera que los datos se transmiten en una sola dirección a través del anillo y cada dispositivo actúa como repetidor de señal para mantener la integridad de la señal hasta llegar al computador o dispositivo destino. (Sánchez Rubio et al., 2020, pp. 98,99; Forouzan, 2012)

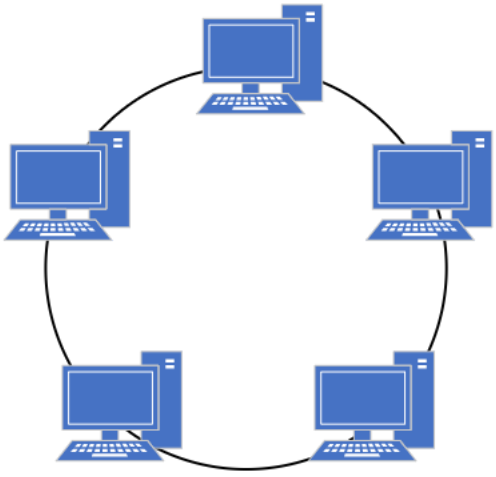


Ilustración 10-2: Topología de red en anillo.

Fuente: (Sánchez Rubio et al., 2020, p. 98)
Elaborado por: Huilcamaygua, Cinthya, 2023.

Sin embargo, una de las limitaciones de esta topología es la necesidad de un dispositivo de control de acceso al medio (MAC) para evitar conflictos de acceso a la red, de manera que el costo de esta solución es significativo, ya que la placa de red puede llegar a tener un valor igual o mayor que una estación de trabajo para que estas estaciones no tengan conflictos entre sí. Otra desventaja es que, si una de las estaciones de trabajo cae, la red deja de funcionar. A pesar de todo esto aún se siguen utilizando en la actualidad debido a su resistencia a las interferencias y su capacidad para soportar grandes cantidades de tráfico de datos como en redes MAN y WAN. Además, el dispositivo servidor puede ser cualquiera de las estaciones de trabajo (Sánchez Rubio et al., 2020)

- **Topología de red en estrella**

La topología de red en estrella es uno de los diseños más populares en redes de computadoras, en la que todos los dispositivos están conectados a un punto central, conocido como un hub o switch (Ver Ilustración 11-2). Este modelo centralizado facilita la detección y resolución de problemas en la red, ya que si un dispositivo falla, no afecta el funcionamiento de los demás dispositivos conectados a la red. Además, la topología en estrella permite una fácil adición o eliminación de dispositivos en la red, lo que la hace muy escalable. A pesar de estas ventajas también tiene sus desventajas, como el hecho de que si el hub o switch central falla, toda la red se vería afectada y también que tiene una baja transmisión de datos. (Forouzan, 2012; Stallings, 2014b)

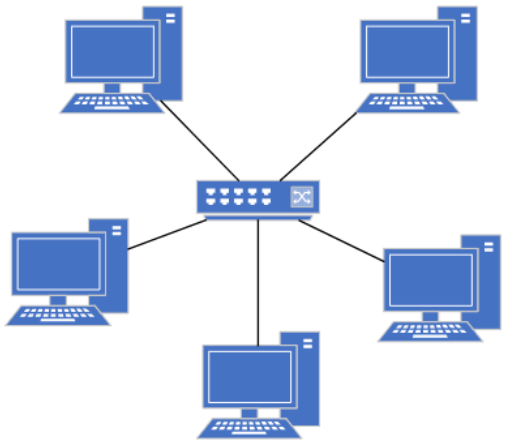


Ilustración 11-2: Topología de red en estrella.
Fuente: (Sánchez Rubio et al., 2020, p. 97)
Elaborado por: Huilcamaygua, Cinthya, 2023.

Este modelo de conexión es fácil de implementar y es escalable, de forma que existen variantes como la topología en estrella extendida, en la que se utilizan varios hubs o switches

interconectados entre sí para crear una red más grande; otra variante es la topología en estrella jerárquica, en la que se utilizan múltiples hubs para segmentar la red en subredes más pequeñas y manejables. (Kurose y Ross, 2017)

- **Topología de red en árbol**

Es una estructura que combina la topología en estrella y la topología en bus. La topología en árbol es común en las redes de área local (LAN) empresariales, donde se conectan varias redes de estrella en un conjunto jerárquico en forma de árbol con sus ramificaciones. En la topología en árbol, un dispositivo de red central, como un switch o un hub, se conecta a los nodos de la red de estrella. Luego, se conectan varios de estos nodos a un dispositivo de nivel superior en la jerarquía, que puede ser otro switch, hub u otro dispositivo de red. La topología en árbol permite la transmisión de datos a alta velocidad y la segmentación de redes grandes en segmentos más pequeños, lo que facilita la administración y el mantenimiento de la red. (Forouzan, 2012; Sánchez Rubio et al., 2020)

Además, esta topología es escalable y permite agregar o quitar nodos sin afectar la red en su conjunto. Sin embargo, una falla en el dispositivo central de la red puede interrumpir la comunicación entre los nodos conectados en la red, tal como se puede observar en la Ilustración 12-2. Por esta razón, se utilizan dispositivos de nivel superior redundantes para garantizar la disponibilidad y la integridad de la red. (Tanenbaum y Wetherall, 2012)

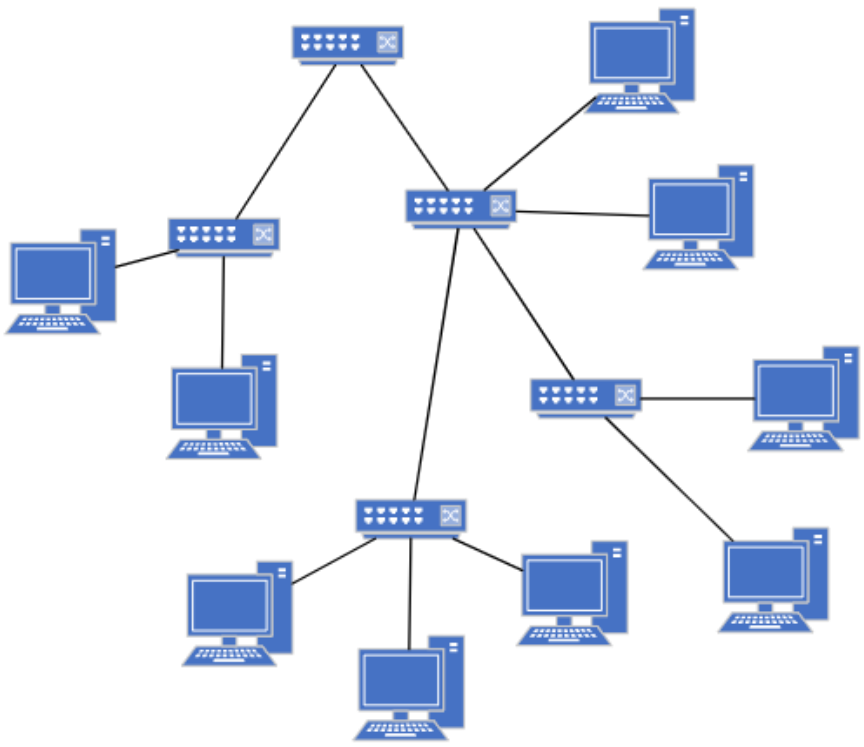


Ilustración 12-2: Topología de red en árbol

Fuente: (Singh y Kumar, 2018, p. 328)

Elaborado por: Huilcamaygua, Cinthya, 2023.

- Topología de red en Malla

Esta topología es una estructura o configuración física en la que cada nodo está conectado directamente a todos los demás nodos de la red. Esta topología proporciona una alta redundancia y confiabilidad en la red, ya que la comunicación se puede establecer de forma directa entre cualquier par de nodos. Además, la topología de red en malla es muy escalable y puede adaptarse a las necesidades cambiantes de la red sin interrupciones importantes. Sin embargo, esta topología requiere una gran cantidad de cableado y puertos de red, lo que puede hacer que sea costosa de implementar en grandes redes. (Tanenbaum y Wetherall, 2012; McMillan, 2015)

Existen dos tipos de esta configuración: la topología de malla completa y la topología de malla parcial. En la topología de malla completa, cada nodo está conectado directamente a todos los demás nodos (Ver Ilustración 13-2), mientras que, en la topología de malla parcial, solo algunos nodos están conectados directamente a otros nodos en la red. En una topología de malla parcial, los nodos que no están conectados directamente a otros nodos pueden acceder a ellos a través de nodos intermedios. La topología de red en malla se utiliza comúnmente en aplicaciones críticas, como la industria aeroespacial y militar, donde se requiere alta redundancia y confiabilidad en la red. (Sánchez Rubio et al., 2020; Forouzan, 2012)

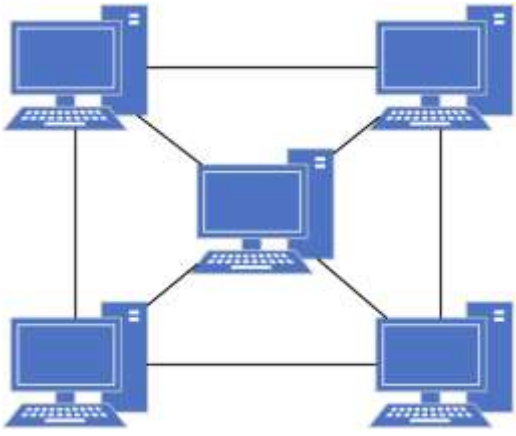


Ilustración 13-2: Topología de red en malla.

Fuente: (Patterson y Hennessy, 2011, p. 622)

Elaborado por: Huilcamaygua, Cinthya, 2023.

- **Topología de red híbridas**

Estas redes han aparecido por la necesidad de expandirse, las topologías hibridas son una combinación de dos o más topologías de red como variantes de una de las configuraciones antes mencionadas: estrella extendida, por ejemplo. Otro ejemplo es una red que combina una topología de estrella y una topología de bus se convierte en una red hibrida. Esta combinación de topologías puede ofrecer beneficios como la redundancia de la red y la capacidad de manejar un gran número de nodos. Sin embargo, también puede ser más costosa y compleja de implementar y mantener.(Singh y Kumar, 2018)

Las topologías de red híbridas se están convirtiendo cada vez más populares en redes de computadoras debido a su capacidad para combinar las fortalezas de diferentes topologías mejorando la eficiencia y la confiabilidad de la red, como se puede ver en la Ilustración 14-2 que combina diferentes tipos de topologías en una sola. En las redes de gran tamaño se requieren de alta redundancia para asegurar que la red esté disponible en todo momento, incluso si falla un componente de la red, llegando a ser las topologías hibridas tolerantes a fallos. Cabe recalcar que en estas redes se pueden mejorar la velocidad y rendimiento de la red, reduciendo el tráfico y optimizando la transmisión de los datos entre los nodos de la red. (Tanenbaum y Wetherall, 2012; McMillan, 2015, p. 97)

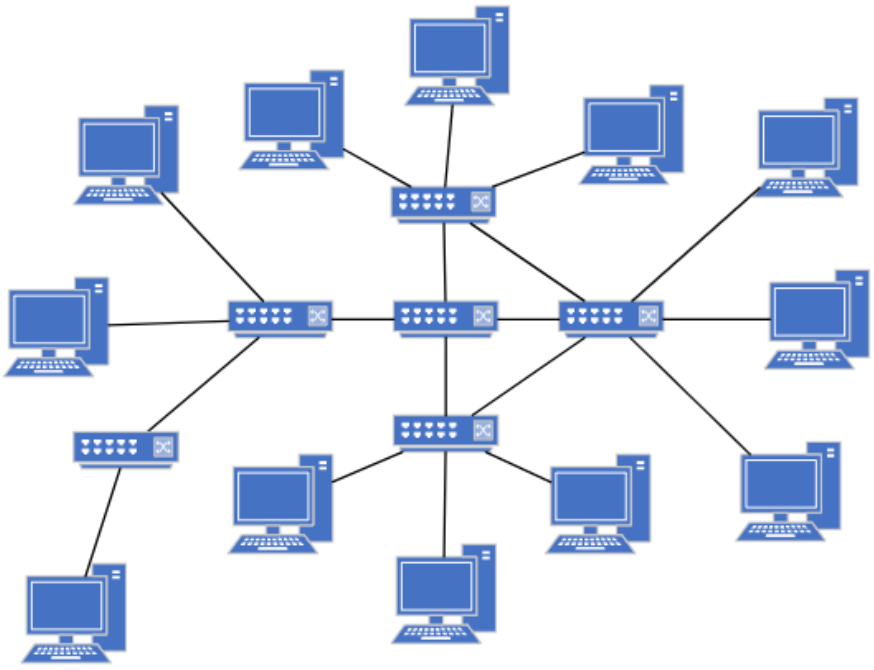


Ilustración 14-2: Topología de red híbrida.

Elaborado por: Huilcamaygua, Cinthya, 2023.

2.2.1.7 Arquitecturas o modelos de comunicación

Desde el surgimiento de las redes de computadores, la forma en cómo se conectaban los ordenadores era por medio de un cable conectados entre sí y donde compartían sus recursos, pero con el pasar del tiempo esta forma de conexión no abastecía a todos los computadores compartir sus recursos de manera eficaz, de manera que debía haber un computador lo suficientemente rápido y que contenga la mayor cantidad de datos que pueda compartir, surgiendo así el Servidor. (Raya Cabrera, 2015, p. 18) Con este nuevo elemento en una red de computadores, este servidor tiene la característica de abastecer las necesidades de los ordenadores demandantes de información, surgiendo la arquitectura Cliente/Servidor.

- **Arquitectura Cliente/Servidor**

El modelo de comunicación cliente/servidor es un ambiente donde participa un equipo cliente que puede ser un PC tradicional o una estación de trabajo cuya interfaz es amigable con el usuario final, y un equipo servidor que puede estar alojado en la nube o una estación física con gran capacidad de procesamiento. En síntesis, comunicación entre cliente y servidor. (Raya Cabrera, 2015, p. 19)

El servidor provee de un servicio o conjunto de servicios a un usuario, estos servicios pueden ser para almacenar (servidores de bases de datos), para ver una película (servidores multimedia bajo demanda), entre otras; destacándose los servidores de bases de datos. (Stallings, 2012)

Un dato interesante sobre este modelo de comunicación es que para atender a un gran número de solicitudes y manejar gran cantidad de ordenadores clientes, utilizan el modelo informático llamado computación distribuida, en donde se utiliza un software dedicado a resolver una cantidad masiva de peticiones, procesándolas a partir de la división de tareas entre los computadores de un mismo nodo. Toda esta estructura de cliente, servidor, y sus respectivas aplicaciones, se encuentran dentro de una red, que puede ser LAN, cuando se crea un servidor virtual local; una red WAN, cuando el servidor que otorga el servicio está dentro del país; o la Internet cuando se adquiere un servidor virtual situado en otro continente para realizar cloud computing, como se puede ver en la Ilustración 15-2. (Stallings, 2012)

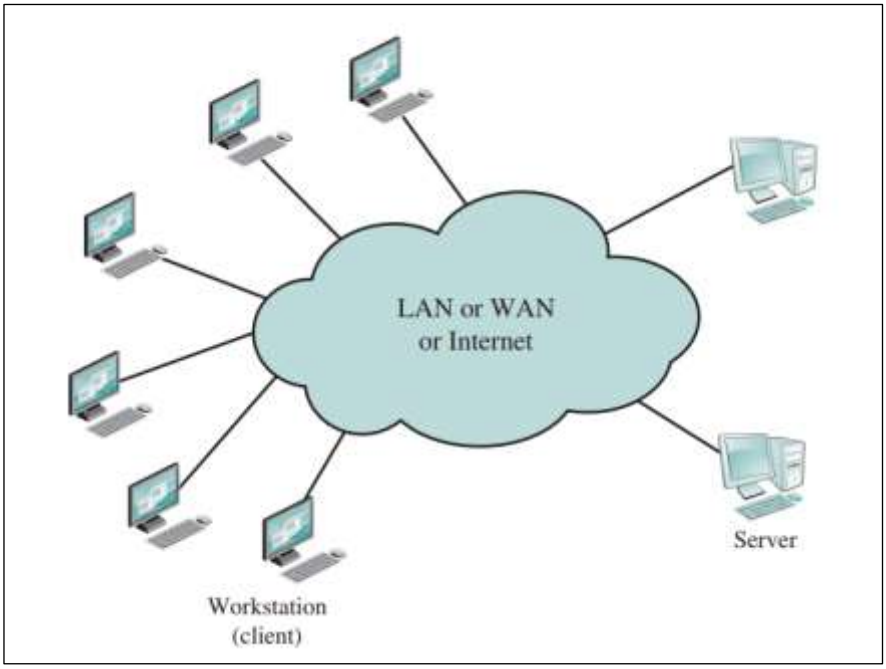


Ilustración 15-2: Diagrama genérico de la comunicación cliente/servidor por las diferentes redes.

Fuente: Stallings, 2012, p.679

- **Arquitectura Publicación/Suscripción**

Este modelo de comunicación se deriva del modelo cliente/servidor. (Stallings, 2012). Se trata de transmitir un mensaje a un destinatario, pero con la diferencia que, en este modelo el que media los mensajes solo lleva la información al destino y no le interesa ni los orígenes del mensaje, ni demasiada información del destinatario.(IBM Corporation, 2015b)

Aquí los elementos que intervienen son, el mensaje, el que envía el mensaje o publicador, el que recibe el mensaje o suscriptor y el que gestiona la comunicación o intermediario y un tópico (Ver Ilustración 16-2). En este modelo, el intermediario solo trata de llevar la información de un lugar a otro, sin mayor esfuerzo por la gestión de estos mensajes puesto que el intermediario solo reconoce tópicos y el dispositivo que publique en nombre de un tópico envía la información hasta el intermediario y lo envía hacia el suscriptor, el dispositivo que este suscrito a un tópico o tema, recibe el mensaje sin importar quien lo envió. Lo más importante para crear esta interacción entre publicadores y suscriptores es que se encuentren conectados hacia un mismo intermediario bajo un mismo tópico. (IBM Corporation, 2015b)

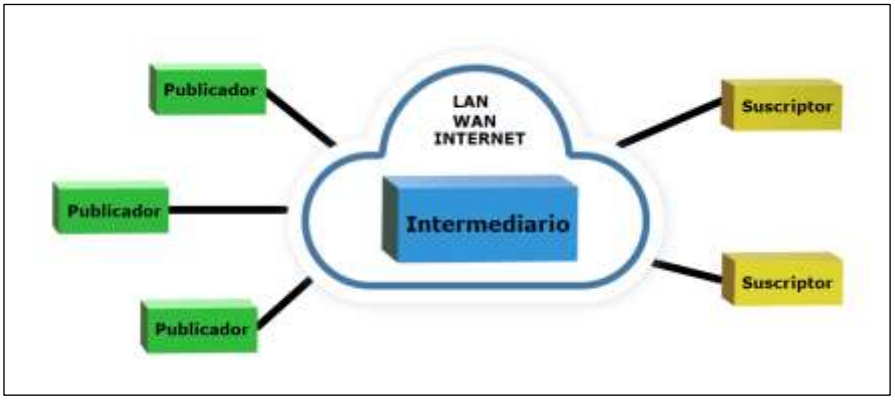


Ilustración 16-2: Diagrama de comunicación para el modelo publicación/suscripción.

Realizado por: Huilcamaygua, Cinthya, 2023.

2.2.1.8 Modelos de referencia o Arquitectura de protocolos

La arquitectura de protocolos se refiere a un conjunto de protocolos y estándares organizados en capas lógicas para facilitar la comunicación entre dispositivos en una red de ordenadores. En esencia, se trata de una estructura en la que cada capa se encarga de una tarea especifica en el proceso de comunicación y transmisión de datos. (Tanenbaum y Wetherall, 2012, p. 35)

Existen varios modelos de referencia al momento de transmitir la información, modelos estandarizados por distintas organizaciones, de las cuales las más utilizadas son el modelo OSI y el modelo de capas TCP/IP. (Sánchez Rubio et al., 2020, p. 23)

- **Modelo OSI**

El modelo de arquitectura de protocolos OSI (Open Systems Interconnection), desarrollado por la Organización Internacional de Estándares (ISO), divide el proceso de comunicación en siete capas lógicas, cada una de las cuales tiene un conjunto de funciones específicas que se encargan de la comunicación de datos entre dispositivos. Cada capa del modelo OSI se comunica con las capas adyacentes mediante protocolos y servicios específicos, y cada capa es independiente de las demás capas del sistema. Esto significa que una capa puede ser modificada sin afectar las demás capas del sistema, lo que permite una mayor flexibilidad y facilidad de mantenimiento. (Day y Zimmermann, 1983)

La arquitectura de protocolos se divide en siete niveles, diseñados para cumplir ciertos principios. En primer lugar, cada nivel se crea para situaciones en las que se requiere un nivel diferente de abstracción, y cada nivel tiene una función bien definida. Además, la función de cada nivel se

elige pensando en la definición de los protocolos estandarizados, y se debe seleccionar la frontera de cada nivel minimizando el flujo de información entre interfaces. También es importante que el número de niveles sea suficientemente grande para evitar la mezcla de funciones diferentes en el mismo nivel, pero lo suficientemente pequeño para que la arquitectura sea fácil de manejar. (Day y Zimmermann, 1983; Sánchez Rubio et al., 2020, p. 96)

En la Ilustración 17-2, se puede observar el orden de las capas del modelo OSI, cada capa se comunica de forma bidireccional con su capa vecina, y a medida que se efectúa la comunicación, los datos pasan por cada una de las capas a modo de etapas.(Sánchez Rubio et al., 2020, p. 96)

En el modelo OSI las capas siguen un orden que van desde el nivel superior representado por la capa de aplicación hasta la capa de nivel inferior o capa física y cada una de estas capas tiene una función específica y se comunican con las capas adyacentes a través de interfaces estándar. En la capa de aplicación, se proporcionan servicios para la comunicación entre aplicaciones. La capa de presentación se encarga de la representación de los datos para que puedan ser interpretados por la aplicación. La capa de sesión establece, mantiene y finaliza sesiones entre aplicaciones. En cuanto a la capa de transporte, su función principal es la transferencia de datos de extremo a extremo a través de la red. La capa de red proporciona servicios para el enrutamiento de paquetes a través de la red. Por otro lado, la capa de enlace de datos se encarga de la transmisión fiable de paquetes entre nodos adyacentes de la red. Por último, la capa física define las especificaciones físicas necesarias para la transmisión de datos a través del medio de comunicación que puede ser cableado o inalámbrico (Marsic, 2013, p. 21). Esto se puede apreciar en la Ilustración 17-2.

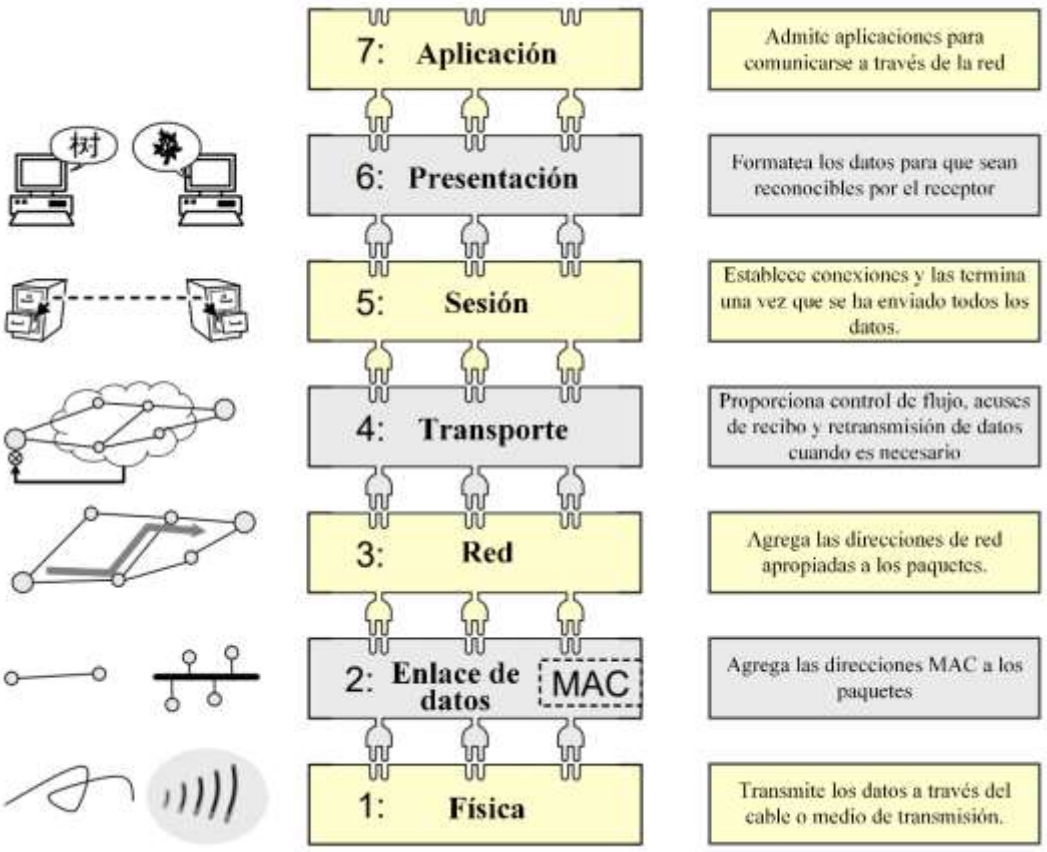


Ilustración 17-2: Funciones de las capas del modelo OSI.

Fuente: (Marsic, 2013, p. 22)

Realizado por: Huilcamaygua, Cinthya, 2023.

Los datos deben pasar por cada una de las capas del modelo de referencia OSI, por lo que empiezan a generarse desde la capa superior hasta llegar al nivel más bajo. En la Ilustración 18-2, se puede apreciar cómo se transmite un dato generado en un computador hasta otro. En esta red, los datos se transmiten desde un nivel superior a uno inferior, y en el extremo opuesto se transmiten desde un nivel inferior a uno superior. Los enrutadores al ser equipos de capa 3, los datos llegan hasta ese nivel, es decir, reciben los datos a través de los cables o capa física, y suben hasta el nivel 3 o capa de red, y así mismo sucede cuando pasa hacia otro enrutador hasta llegar al ordenador destino. (Marsic, 2013, p. 20; Sánchez Rubio et al., 2020, p. 23)

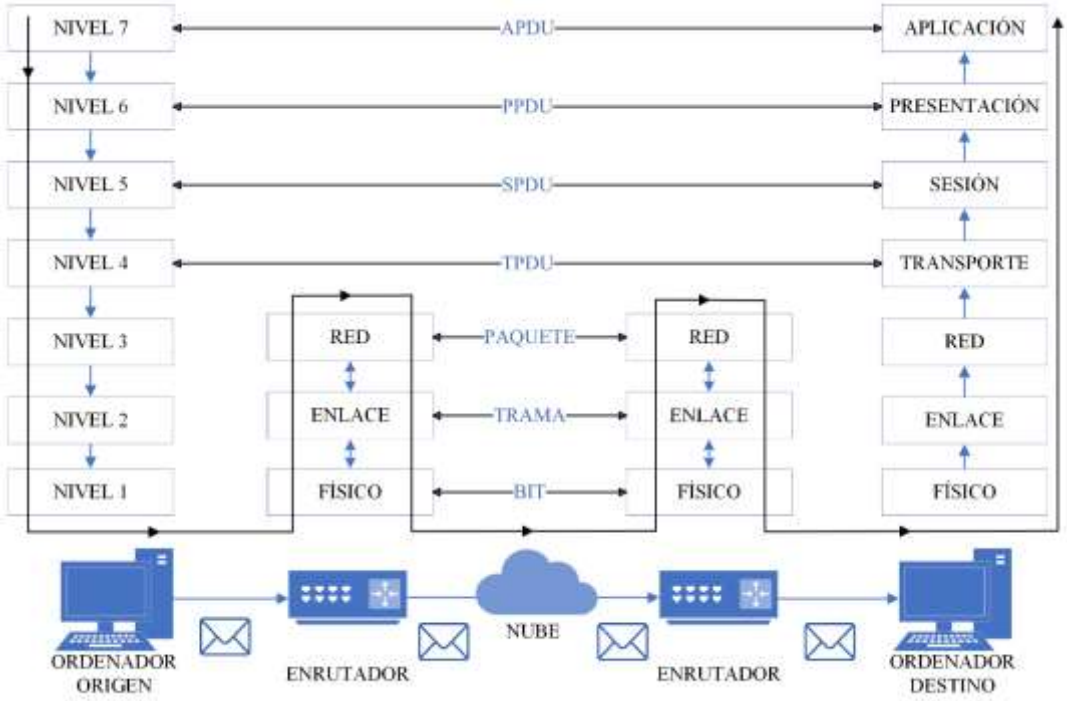


Ilustración 18-2: Transmisión de datos a través de las capas del modelo OSI

Fuente: (Sánchez Rubio et al., 2020, p. 23)

Realizado por: Huilcamaygua, Cinthya, 2023.

Para que esto sea posible, los niveles homólogos de las dos máquinas deben comunicarse entre sí, lo que se logra mediante la inclusión de cabeceras en los paquetes de datos. Estas cabeceras contienen información necesaria para el control de la información, como el sincronismo, el control de flujo y la corrección de errores. Es importante destacar que los datos que no son parte de las cabeceras deben ser transparentes a los niveles, es decir, deben ser transmitidos al siguiente nivel sin importar su contenido. De esta manera, se asegura una transmisión eficiente y efectiva de datos a través de la red. (Sánchez Rubio et al., 2020, p. 23)

En la Ilustración 19-2, se puede observar con más detalle el proceso por el que pasan los datos para ser comunicados hacia otro computador, cada capa añade una cabecera a los datos que vienen desde su capa adyacente superior y, ese conjunto de cabecera y datos conforman una unidad de información que puede ser bit, trama, paquete, y los PDU (Unidad de Datos de Protocolo). Por dar un ejemplo, en la capa de aplicación se coloca la cabecera de capa de aplicación (CA); el conjunto de CA y datos son llamados APDU (PDU de la capa de aplicación). En el caso de la capa de red, enlace y capa física, la unidad de información es llamada paquete, trama y bits respectivamente. A este proceso de colocar cabeceras a medida que pasan los datos a través de las capas se llama encapsulación de datos.

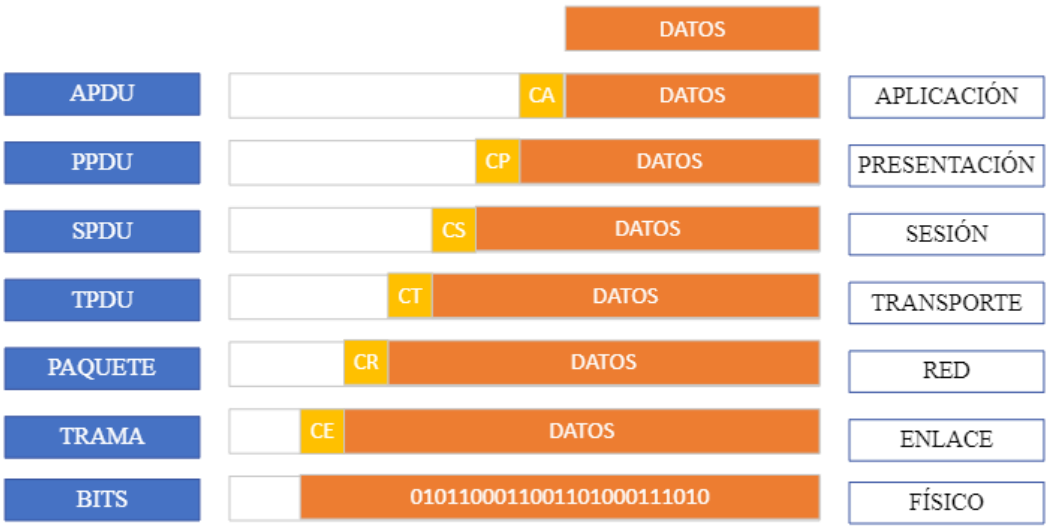


Ilustración 19-2: Presentación de datos por capa del modelo de referencia OSI.

Fuente: (Sánchez Rubio et al., 2020, p. 26)

Realizado por: Huilcamaygua, Cinthya, 2023.

La encapsulación de datos envuelve a los datos originales en capas adicionales de encabezados a medida que descienden por el stack de protocolos[4]. Pues, según el protocolo utilizado del stack de protocolos del Modelo OSI, el mensaje puede tener mayor o menor peso al efectuarse la comunicación a través de los distintos dispositivos que conforma una red de ordenadores. (Sánchez Rubio et al., 2020, p. 23; Tanenbaum y Wetherall, 2012, p. 27)

- **Modelo TCP/IP**

El Modelo TCP/IP, que consta de cuatro capas: la capa de aplicación, la capa de transporte, la capa de Internet y la capa de acceso a la red. Este modelo se utiliza ampliamente en Internet y en muchas redes empresariales. (Aznar López, 2005, p. 21)

Este modelo no utiliza al modelo OSI como una referencia, debido a que el modelo TCP/IP fue desarrollado antes que el modelo OSI usando el modelo de referencia del DoD (Departamento de defensa). Si bien tiene muchas similitudes con el modelo OSI, se puede decir que es una versión simplificada con sus 4 capas, como se ilustra en la Ilustración 20-2. Y como su nombre lo indica utiliza literalmente los protocolos TCP (Protocolo de control de transmisión) e IP (Protocolo de internet). (Blank, 2004, p. 24)

[4] El stack de protocolos se refiere a una lista de protocolos que se utilizan en cada capa del modelo de referencia

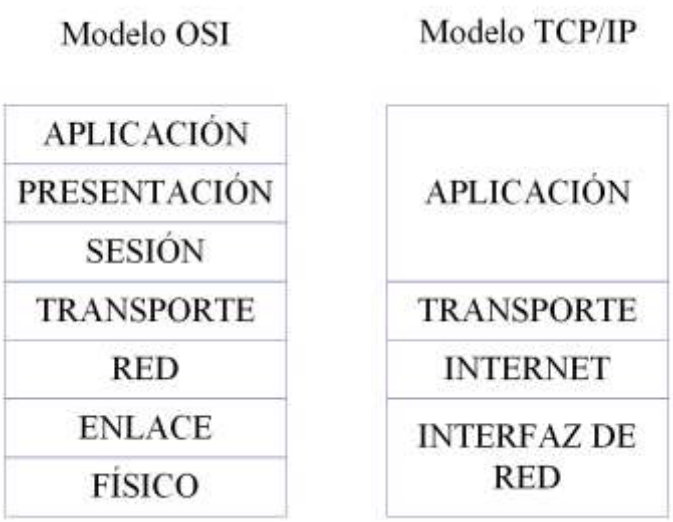


Ilustración 20-2: Modelos de referencia OSI y TCP/IP

Fuente: (Aznar López, 2005, p. 21)

Realizado por: Huilcamaygua, Cinthya, 2023.

Las capas del modelo TCP/IP cumplen diferentes funciones, pero estas funciones son semejantes y se asocian a las funciones del modelo OSI, por lo que algunas capas de modelo OSI se pueden ver contenidas o incluidas dentro del modelo TCP/IP. (Aznar López, 2005)

2.2.2 Bases de datos

2.2.2.1 Introducción a las bases de datos

En todo el mundo ha existido la necesidad de almacenar información para buscarla posteriormente y analizarla. La humanidad desde tiempos antiguos ha tratado de guardar la información y revisarla en caso de requerirlo en grandes bibliotecas como la conocida biblioteca de Alejandría del antiguo Egipto. Esto puede significar un indicio de las bases de datos, y en definitiva las bases de datos son una gran biblioteca con información digital. (Dueñas Nogueras, 2023, p. 191)

Antes del surgimiento de las bases de datos, en la década de 1960, existieron los sistemas de ficheros, sistemas que en su tiempo fueron de mucha ayuda frente a la búsqueda de información de forma manual, pero al seguir un modelo descentralizado, los datos que se generan en una dependencia o departamento de una empresa son diferentes de otra, y datos que eran comunes entre dependencias se duplicaban, generando inconsistencias. De manera semejante ocurre cuando se necesitaban sincronizar los datos, tarea que para el programador era tedioso y a la empresa le incurría en perdida de dinero y tiempo, sin quedar exentos de que se generen aún más errores (Navathe y Elmasri, 2007a; López Querol et al., 2023)

A partir de esta problemática nace el concepto de bases de datos, un programa que se crea y define una sola vez, que puede ser compartido entre varios usuarios pudiendo tener información con común sin duplicados y actualizados. (López Querol et al., 2023, p. 273)

Las bases de datos han evolucionado notablemente, y en la actualidad juegan un papel importante que se pueden encontrar en prácticamente todos los ámbitos de la vida cotidiana desde la gestión de información en empresas y organizaciones, hasta realizar acciones comunes como una transferencia bancaria, un mensaje en aplicaciones de mensajería. Las bases de datos son herramientas fundamentales para el manejo de información en la era digital. (Navathe y Elmasri, 2007)

2.2.2.2 Definición

Una base de datos es un almacén donde se guardan datos para utilizarlos posteriormente para analizarlos, esto por definirlo de una forma sencilla (Navathe y Elmasri, 2007).

En la obra de Marqués, se define a una base de datos como "un conjunto de datos almacenados en memoria externa que están organizados mediante una estructura de datos" (Marqués, 2011a). Mientras que en el libro Database systems, una base de datos se define como "un conjunto de datos interrelacionado, almacenados en una estructura de manera que sea accesible para su consulta y actualización" (Connolly y Begg, 2015, p. 5)

Estas dos definiciones destacan la importación de la estructuración y la accesibilidad de los datos para su correcta gestión en una base de datos.

2.2.2.3 Tipos de bases de datos

Las bases de datos se componen de diferentes elementos, entre los cuales se incluyen tablas, campos, registros y claves. Las tablas contienen los datos organizados en columnas y filas, y los campos corresponden a las columnas que representan los diferentes tipos de datos que se almacenan. Los registros, por su parte, son las filas que contienen los datos individuales, y las claves se utilizan para identificar de manera única cada registro en la base de datos (Connolly y Begg, 2015, p. 5). Además, existen diferentes tipos de bases de datos según su organización, como las bases de datos relacionales o SQL, las bases de datos NoSQL y las bases de datos orientadas a objetos.(López Querol et al., 2023, p. 252)

- **Bases de datos relacionales**

Son bases de datos muy utilizadas pues fueron las primeras en aparecer. Se llaman bases de datos relacionales o SQL (por Structured Query Languaje) a sistemas que utilizan una estructura de datos basadas en tabla, o dicho técnicamente, siguen un esquema entidad-relación donde asignan a los datos de acuerdo con sus características a una fila y columna de una tabla. Un modelo relacional sigue una estructura lógica del concepto de relación (Marqués, 2011b)

En esta base de datos con forma de tabla, también participa un elemento llamado clave primaria, esta llave primaria puede estar formada por varias columnas de una tabla, es necesaria para establecer un vínculo entre tablas, pues cada tabla deberá tener una clave que haga referencia a la clave de otra tabla. (Marqués, 2011a)

En fin, las bases de datos relacionales tienen varias características importantes. En primer lugar, se basan en un modelo relacional y utilizan el lenguaje SQL para gestionar los datos. Además, estas bases de datos están diseñadas para escalar verticalmente y se utilizan para almacenar datos con un esquema predefinido. Todos los datos almacenados deben seguir la misma estructura, lo que puede hacer que la modificación de datos sea compleja y perjudicial para el sistema. Las bases de datos relacionales requieren de grandes máquinas con buen nivel de rendimiento para su ejecución y son ideales para sistemas de contabilidad debido a su eficiencia en la consulta de datos estructurados y transacciones de varias filas. (López Querol et al., 2023, pp. 250,251)

- Bases de datos no relacionales

Este tipo de bases de datos resultó de la búsqueda de una solución a problemas de gestión generados por las bases de datos relacionales, orillando a crear una solución robusta llamada bases de datos no relacionales o NoSQL. (Acens, 2014)

Los problemas más grandes generados en las bases de datos SQL es el rendimiento y la escalabilidad, ya que miles de personas que realizan consultas diariamente afecta en el rendimiento conllevando a problemas de escalabilidad porque no admitiría muchos usuarios y las búsquedas se volverían exhaustivas para la base de datos. (Acens, 2014)

Estas bases de datos NoSQL, como su propio nombre lo dice, no se basan en el modelo de entidad-relación o relacional y tampoco ubican a los datos en una tabla. La forma de almacenar los datos es distinta, usan otras técnicas como la de clave-valor, documentales, en grafos, y orientada a objetos. (López Querol et al., 2023, p. 264)

Las virtudes de las bases NoSQL se basan en las falencias de las relacionales, en primer lugar, no necesita de un equipo con muchos recursos y por ende se reducen los costes de su implementación, la escalabilidad es horizontal con la adición de más nodos, manejan una gran cantidad de datos sin generar cuellos de botella porque utilizan una estructura distribuida y no centralizada en una sola máquina.(Acens, 2014b; Sadalage y Fowler, 2013)

En definitiva, las bases de datos no relacionales se caracterizan por presentar ciertas particularidades. En primer lugar, no se basan en un modelo relacional y están diseñadas para escalar horizontalmente. Estas bases de datos son ideales para almacenar datos no estructurados, ya que permiten que la estructura de los datos varíe y que cada documento pueda tener su propia estructura. Además, es posible realizar cambios en la base de datos sin detener el sistema y sin alterar el funcionamiento de esta. Las bases de datos no relacionales pueden ejecutarse en máquinas con pocos recursos, que en conjunto pueden lograr un rendimiento potente. Asimismo, la optimización de las consultas se enfoca en grandes cantidades de datos o en bases de datos que cambian constantemente. (Sadalage y Fowler, 2013)

2.2.2.4 Diseño de bases de datos

El diseño de bases de datos implica la identificación de las entidades y relaciones relevantes en el dominio de la aplicación y la creación de una estructura de datos que represente esas entidades y relaciones de manera efectiva. Esto incluye la definición de tablas, campos y restricciones de integridad para garantizar la coherencia y consistencia de los datos almacenados. También puede incluir la identificación de las consultas y procesos necesarios para recuperar, insertar, actualizar y eliminar datos de la base de datos. (López Querol et al., 2023, pp. 233,235)

Debido a que las bases de datos son el medio principal por el cual las aplicaciones almacenan y recuperan información, su diseño es esencial para el desarrollo de aplicaciones. Un diseño eficaz de la base de datos puede aumentar el rendimiento de las aplicaciones, garantizar la integridad de los datos y facilitar el mantenimiento y el desarrollo de aplicaciones a lo largo del tiempo.(Harrington, 2016)

- **Modelos de Datos:**

Se exploran diferentes modelos de datos, como el modelo relacional, el modelo de entidad-relación, el modelo jerárquico y el modelo de red. Cada modelo ofrece un enfoque diferente para organizar y representar los datos.

- **Normalización:**

Se discuten las formas normales, que son reglas de diseño utilizadas para eliminar redundancias y anomalías en una base de datos relacional. La normalización es esencial para garantizar la integridad y la consistencia de los datos.

- **Relaciones:**

Se abordan las relaciones entre las tablas o entidades en una base de datos relacional. Esto incluye conceptos como las claves primarias, las claves foráneas y la definición de relaciones uno a uno, uno a muchos y muchos a muchos.

- **Diseño Físico:**

Se explora cómo se implementan las estructuras de datos en un sistema de gestión de bases de datos (DBMS) específico. Esto incluye decisiones sobre la asignación de datos en bloques de almacenamiento, la creación de índices para mejorar el rendimiento y otras consideraciones técnicas.

- **Optimización:**

Se considera cómo optimizar el diseño de la base de datos para mejorar el rendimiento de las consultas y las operaciones. Esto puede involucrar la elección de índices adecuados, la desnormalización selectiva y otras estrategias.

- **Integridad y Seguridad:**

Se discute cómo se pueden implementar reglas de integridad y medidas de seguridad en el diseño de la base de datos para garantizar que los datos sean precisos y estén protegidos.

2.2.3 Virtualización

2.2.3.1 Introducción a la virtualización

La virtualización es una tecnología cada vez más importante en el mundo de la informática y las tecnologías de la información. En términos simples, la virtualización permite crear múltiples entornos de software que se ejecutan en una sola máquina física. Esto puede ayudar a maximizar

el uso de los recursos de hardware, reducir los costos y mejorar la eficiencia. (Smith y Nair, 2005, p. 32)

La importancia de la virtualización radica en la reducción de costes tanto de enfriamiento como de mantenimiento y modernización (o renovación) de los equipos servidores; esto se debe a que, antes de que existiese el concepto de virtualizar, se adquiría equipamiento individual para cada servicio como el correo, web, http, bases de datos, etc. y que posteriormente tendrían que ser reemplazadas con nuevos equipos lo que implica costes elevados en una empresa. (Sepúlveda Rodríguez et al., 2022; Kusnetzky, 2011)

Este concepto se remonta a los antiguos mainframes, que debían ser divididos para varios usuarios en entornos de aplicación completamente diferentes. Esa realidad de la década de 1970 fue en gran medida superada en 1980 y 1990, con el surgimiento de los ordenadores personales. (González Rio, 2014)

Han pasado aproximadamente 50 años desde que apareció la tecnología de virtualización, ha resurgido en la década de los 90 y en estos últimos años ha tenido un gran despunte por la gran era tecnológica en la que vivimos, actualmente existe gran cantidad de servicios y aplicaciones que para ofrecer sus servicios requieren de un servidor, servidor el cual ha sido virtualizado para que coexista con otros servicios de forma independiente dentro de un mismo equipo servidor físico optimizando la utilización del recurso hardware. (Pessolani et al., 2012; González Rio, 2014)

La virtualización puede ser utilizada en una amplia variedad de situaciones, desde servidores de empresas hasta entornos de prueba y desarrollo. Además, existen varias tecnologías de virtualización diferentes disponibles, cada una con sus propias ventajas y desventajas. (Sepúlveda Rodríguez et al., 2022)

2.2.3.2 Definición

El término virtualización tiene diversas definiciones, pero la idea es la misma en todas ellas. Red Hat se refiere a la virtualización como "una tecnología que permite crear múltiples entornos simulados o recursos dedicados desde un solo sistema de hardware físico" (Red Hat, 2018). Permite utilizar la capacidad total de una máquina física distribuyendo sus capacidades entre muchos usuarios o entornos[5]. (Red Hat, 2017)

[5] Traducida del inglés de la página web de Red Hat

Asimismo, un grande en el tema, VMware define a la virtualización como:

> *La virtualización consiste en crear una representación basada en software, o virtual, de una entidad física como, por ejemplo, aplicaciones, servidores, redes y almacenamiento virtuales… La virtualización utiliza el software para imitar las características del hardware y crear un sistema informático virtual, para ejecutar más de un sistema virtual, y múltiples sistemas operativos y aplicaciones, en un solo servidor* (VMware, 2020)

Al igual que los dos anteriores, González se refiere a la virtualización de manera concisa "la virtualización separa las funciones de hardware permitiendo que en una misma maquina sean ejecutadas simultáneamente dos o más entornos diferentes y aislados".(González Rio, 2014)

2.2.3.3 Elementos de virtualización

Según las definiciones descritas en el punto anterior, la virtualización se trata de compartir los recursos de hardware y distribuirlos de manera individual entre diferentes usuarios para así aprovechar toda la capacidad del equipo físico.

El servidor físico sin virtualización se compone de 3 elementos básicos como: el hardware, el sistema operativo (software) y las aplicaciones sobre este sistema operativo (software); como se puede ver en la Ilustración 21-2.

Ilustración 21-2: Componentes de un servidor sin virtualizar.

Fuente: (Villar Fernández y Gómez, 2010, p. 26)
Realizado por: Huilcamaygua, Cinthya, 2023.

Para empezar a virtualizar un equipo informático, primero debemos conocer los términos usados en la virtualización como hipervisor y máquina virtual, elementos que juegan un rol importante, para un mejor entendimiento de temas posteriores a este. Tanto la máquina virtual como el hipervisor dependen el uno del otro.

- **Máquina Virtual**

O también llamado Virtual Machine (VM) en inglés, la máquina virtual no es más que un software que contiene un entorno totalmente aislado, muy aparte del sistema de la maquina física, donde se incluye un sistema operativo y una aplicación; que, al ser instalada en el ordenador, accede a los recursos de hardware y se comporta como una maquina autónoma, pudiendo así ejecutar varios sistemas operativos en un solo host[6]. (VMware, 2020)

Es un sistema de archivos completo que simula ser un computador al ser ejecutada dentro de un equipo físico anfitrión o host, utilizando las capacidades físicas de este último.

La diferencia de la máquina virtual con una maquina física no es el sistema operativo ni nada por el estilo, sino que la máquina virtual está basada netamente en software y "cree" que es una maquina física real con elementos físico-reales, cuando en contexto no es así (González Rio, 2014).

Como se puede ver en la Ilustración 22-2, está la estructura de un servidor físico virtualizado, en donde se encuentran 2 máquinas virtuales y que se pueden extender en número hasta n máquinas virtuales, hasta donde alcancen los recursos del equipo físico. Estas máquinas virtuales aparentemente cuentan con un hardware, sistema operativo y aplicaciones físicos, pero todas ellas tienen aquellos elementos virtualizados gracias a una capa de virtualización. De manera que cada máquina virtual tiene su propia dirección MAC, como un equipo físico. (Pessolani et al., 2012)

[6] En este tema la palabra host se refiere a un servidor único, un único servidor físico.

Ilustración 22-2: Máquinas virtuales dentro de un equipo físico, con todos los elementos que conformaría un equipo físico

Fuente: (Villar Fernández y Gómez, 2010, p. 27)

Realizado por: Huilcamaygua, Cinthya, 2023.

- Hipervisor

También conocido como gestor de virtualización o monitor de máquina virtual (VMM), es un software intermediario que tiene kernel propio y facilita la virtualización, se sitúa entre el hardware y el sistema operativo (Ver Ilustración 23-2), separando las funciones de estos dos y tiene como objetivo controlar "la cantidad de accesos que los sistemas operativos y aplicaciones tendrán al procesador y recursos de hardware, como memoria y lectura y escritura de datos". (González Rio, 2014)

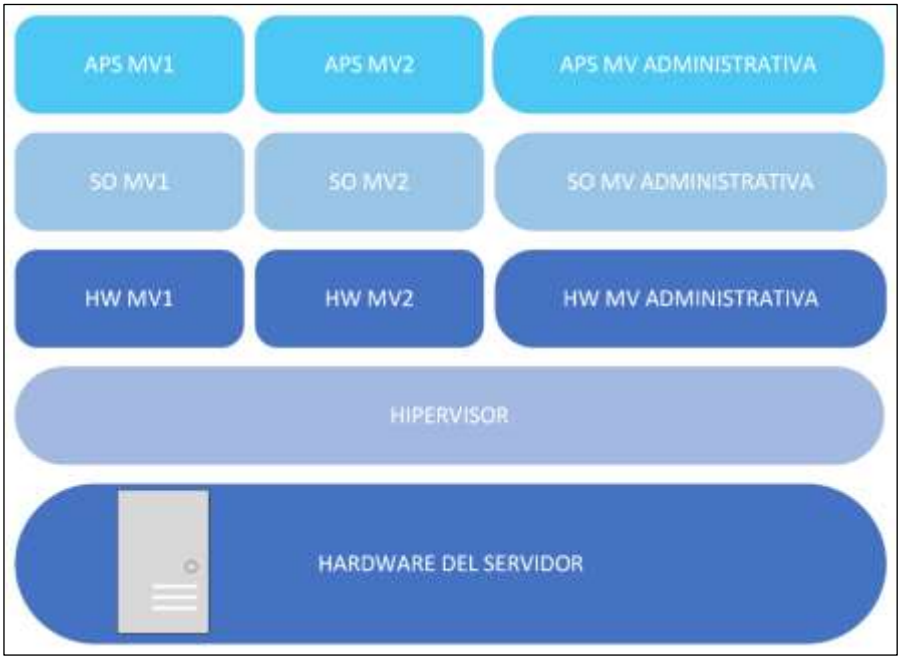


Ilustración 23-2: Ubicación del hipervisor en la arquitectura de virtualización

Fuente: (Villar Fernández y Gómez, 2010, p. 27)

Realizado por: Huilcamaygua, Cinthya, 2023.

Para definir al hipervisor, Villar se refiere de la siguiente forma:

> *El hipervisor o hypervisor es un pequeño monitor de bajo nivel de máquinas virtuales que se inicia durante el arranque, antes que las máquinas virtuales, y que normalmente corre justo sobre el hardware… aunque también lo puede hacer sobre un sistema operativo (llamado en este caso hipervisor hosted)* (Villar Fernández y Gómez, 2010)

Es decir, este software es el elemento que hace posible la virtualización, es el medio por el cual se asigna los recursos de hardware a cada una de las máquinas virtuales que están alojadas dentro de un host.

- **Sistema operativo invitado o huéspedes**

Los sistemas invitados (o OS Guest) denominado también como servidor virtual, se ubican dentro de un sistema operativo Anfitrión o Host, su función es la de compartir un mismo recurso (el núcleo o kernel) con el Host que lo aloja; aunque la máquina virtual también es un huésped de una maquina física, se diferencia del sistema operativo invitado porque la VM si tiene un núcleo propio mientras que el servidor virtual no, por lo que esta última resulta ser de ejecución ligera a comparación de una máquina virtual. (Pessolani et al., 2012)

- Sistema operativo anfitrión o Host

En termino de sistemas operativos, este host es el que aloja a un OS Guest y tiene una única instancia o un único entorno. (Sepúlveda Rodríguez et al., 2022)

2.2.3.4 Ventajas de la virtualización

La virtualización ha transformado la forma en que las organizaciones gestionan sus recursos tecnológicos y ofrece una serie de ventajas notables en diversos campos. En primer lugar, permite la consolidación de servidores físicos, lo que resulta en una utilización más eficiente de los recursos y una reducción significativa en los costos de hardware y energía. Además, la virtualización mejora la flexibilidad al permitir la rápida implementación y migración de máquinas virtuales, lo que se traduce en una mayor agilidad para atender las demandas cambiantes del negocio. (Pessolani et al., 2012)

Otra ventaja es la facilidad de administración. Las máquinas virtuales pueden ser gestionadas centralmente, lo que simplifica la supervisión y el mantenimiento. Esto reduce la necesidad de intervenciones manuales, disminuyendo los errores humanos y mejorando la fiabilidad. La virtualización también contribuye a la seguridad, ya que las máquinas virtuales están aisladas entre sí, lo que minimiza los riesgos de compromiso de datos sensibles. (Sepúlveda Rodríguez et al., 2022; González Rio, 2014)

La recuperación ante desastres se ve favorecida por la virtualización, ya que las copias de seguridad y la migración a servidores secundarios se vuelven más eficientes y rápidas. Asimismo, la escalabilidad se vuelve más sencilla al agregar recursos virtuales según sea necesario, permitiendo un crecimiento flexible sin grandes inversiones iniciales. (González Rio, 2014)

2.2.3.5 Tipos de virtualización

Al hablar de virtualización de manera general significa que “es el efecto de abstraer los recursos de un computador, proporcionar acceso lógico a recursos físicos” lo que nos da una clara visión de que esos recursos del computador pueden ser distintos como el almacenamiento, servidor, red, y asimismo las técnicas de virtualización toman distinto nombre que se categorizan en 4 grupos o modelos de virtualización: virtualización de plataforma, virtualización de recursos, virtualización de aplicaciones y virtualización de escritorio. (Villar Fernández y Gómez, 2010)

- **Virtualización de plataforma**

El sistema completo es el recurso por abstraerse en esta categoría, se trata de tomar todo el hardware del equipo para particionar sus atributos, asignándose a varias instancias de sistemas operativos de forma exclusiva, según corresponda, para que pueda ser ejecutado como una máquina independiente. Es aquí donde tiene presencia los términos hipervisor y máquina virtual. Ya albergadas las máquinas virtuales en el equipo, cada máquina virtual ve a la otra, no como una VM sino como otra maquina muy aparte, sin conocer que están compartiendo el mismo hardware.(Villar Fernández y Gómez, 2010)

- **Virtualización de aplicaciones**

En este tipo de virtualización nos ayuda a ejecutar aplicaciones o programas en un sistema para el cual no fue diseñado, es decir, ejecuta la aplicación sin importar el sistema operativo, no es igual a la emulación. (Citrix, 2020)

Lo que sucede dentro de este tipo de virtualización es que encapsula a la aplicación y la ejecuta sobre el sistema operativo, la aplicación cree que esta interactuando con el S.O. y con el hardware de forma directa, pero en realidad lo hace mediante una máquina virtual. Hay que tener muy en cuenta que la virtualización se centra netamente en las aplicaciones, mas no en el sistema operativo anfitrión. (Villar Fernández y Gómez, 2010)

- **Virtualización de escritorio**

Consiste en simular la carga (datos, archivos) de una estación mediante el uso de máquinas virtuales, opera de forma remota por lo que la estación no se encuentra ligada a una maquina física, sino que se aloja en un servidor central y este escritorio puede ser visto desde diferentes maquinas físicas como smartphones y computadores sin almacenar datos en el disco duro del dispositivo local. (Villar Fernández y Gómez, 2010; González Rio, 2014)

La página web de Citrix menciona en su glosario que la virtualización de escritorio “permite un acceso más seguro y portátil a los recursos del centro de datos” (Citrix, 2020); es portátil porque todo el escritorio no está ligado a la maquina física, sino a un servidor central que aunque sea desconocido el almacenamiento físico, simplemente se accede, no importa en que dispositivo se acceda a ese escritorio, lo importante es que se puede manipularlo.

Tanto la virtualización de escritorio como de aplicaciones son usadas en gran medida por su portabilidad y disponibilidad de acceso 24/7, pueden acceder sin problemas desde cualquier

equipo físico sin importar el sistema operativo, además no incurren en gastos económicos porque todo se guarda en un servidor centralizado de alguna oficina de TI[7] y el mantenimiento no es tan grande como en estaciones físicas individuales, y también existe la facilidad de ingresar a un entorno como si fuera un entorno original con la misma experiencia y las mismas prestaciones. (Villar Fernández y Gómez, 2010)

2.2.4 Internet de las cosas

2.2.4.1 Introducción al Internet de las cosas

La idea de controlar artefactos a través de una red ha existido desde hace muchos años, antes de que se acuñara el término Internet de las cosas gracias a Kevin Ashton, empezando por el control a través de la red telefónica en los 70s y posteriormente mediante el despliegue de una red dedicada con protocolos propietarios de empresas pertenecientes a la industria. (Rose et al., 2015)

En el año 1999, donde la tecnología inalámbrica ha tenido un gran avance, se usó por primera vez la terminología Internet de las cosas por el informático de nacionalidad británica Kevin Ashton. Internet of Things fue el nombre que le dio a un proyecto que usaba la tecnología inalámbrica RFID a través de Internet con el fin de hallar la localización física de productos de la empresa transnacional donde trabajaba. (Ashton, 2009)

Así es como se dio inicio a una nueva era del Internet, el término entró en vigor en el año 2008 aproximadamente, siendo un tema bastante atractivo para las masas y las empresas. Ahora el término es usado para referirse a todo dispositivo conectado a la red que no sea un computador como la ropa, los zapatos, sensores, electrodomésticos, máquinas, etc. (Ashton, 2009)

Incluso ha existido una evolución de las redes de comunicación de maquina a máquina (M2M) hacia Internet de las cosas como se menciona en el libro "From Machine to Machine to the Internet of Things: Introduction to a New Age of Intelligence". Donde se indica esta transformación o transición que han tenido los sistemas M2M frente a la creciente conectividad de dispositivos y la integración de sensores y dispositivos en Internet. (Lee y Lee, 2015)

2.2.4.2 Definición

[7] Tecnologías de la Información

Tal y como afirma Ashton, aunque él haya sido la primera persona en adoptar esta frase no le da derecho a controlar como otras personas lo definan (Ashton, 2009), es por esa razón que no existe una definición única ya que diferentes organizaciones toman el término y lo describen según su punto de vista y de una forma particular pero con la singular característica en común de objetos y conectividad.

Ashton desde su punto de vista, muestra su percepción de IoT publicado con sus propias palabras en la RFID Journal en el año 2009:

> *"Hoy en día, las computadoras, y, por lo tanto, Internet, dependen casi por completo de los seres humanos para obtener información. […] El problema es que las personas tienen tiempo, atención y precisión limitados, lo que significa que no son muy buenos para capturar datos sobre cosas del mundo real. […] Si tuviéramos computadoras que supieran todo lo que había que saber sobre las cosas, utilizando los datos que recopilaron sin nuestra ayuda, podríamos rastrear y contar todo, y reducir en gran medida el desperdicio, la pérdida y el costo. Sabríamos cuándo las cosas necesitaban ser reemplazadas, reparadas o retiradas del mercado, y si estaban frescas o mejor que nunca […] La tecnología RFID y de sensores permite a las computadoras observar, identificar y comprender el mundo, sin las limitaciones de los datos ingresados por humanos[8]"* (Ashton, 2009)

IoT se define a partir del contexto de la tecnología RFID de Ashton: los equipos computacionales pueden capturar los datos del entorno físico más precisos que los humanos y pueden procesar estos datos sin la intervención del hombre.

Según la RFC 7452 sostiene al término IoT como:

> *Internet de las cosas denota una tendencia en que un gran número de dispositivos embebidos utilizan los servicios de comunicación que ofrecen los protocolos de Internet. A estos dispositivos suelen llamarles "objetos inteligentes'' y no son operados directamente por un ser humano, sino que existen como componentes en edificios o vehículos o están distribuidos en el entorno*. [9](Tschofenig et al., 2015)

En el libro de *Tecnología internet of things (IoT) y el big data* definen a IoT de la siguiente forma:

> *IoT es la interconexión digital de objetos cotidianos con internet. Permite el cambio automático de información con otros dispositivos o centros de control sin intervención humana, capturando gran*

[8] Traducido al español de la página RFID Journal
[9] Traducido al español de la RFC 7452

cantidad de información clave sobre uso y rendimiento. Así, facilita la monitorización y operación, creando experiencias únicas y oportunidades inéditas para personas empresas y ciudades. (Ferney et al., 2019)

El término ha tomado muchos significados, pero llegan a una misma conclusión. IoT es la extensión de la conectividad hacia objetos de la vida cotidiana, sensores y demás "cosas" que en un pasado no se podría haber imaginado que se podrían conectar a la red y que pudieran interactuar entre ellos sin la intervención humana como una verdadera comunicación máquina-máquina o M2M (Lee y Lee, 2015; Al-Fuqaha et al., 2015).

Hoy en día se pueden obtener datos de distintos dispositivos conectados a Internet y gracias a las variadas redes de acceso con la que tiene convergencia, IoT permite abrir un abanico de posibilidades de aplicación en las áreas de la salud, educación, transporte, agricultura, hogar, gobierno e industrias; y con las investigaciones que se vienen desarrollando es posible aumentar las aplicaciones en las cuales se desempeña el IoT (Rose et al., 2015; Barrio, 2020, p. 24). (Ver Ilustración 24-2)

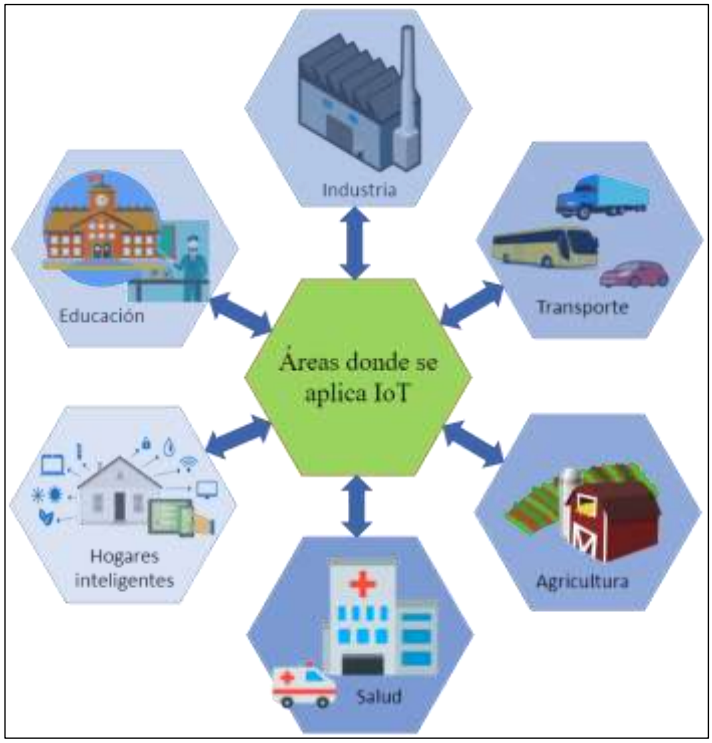


Ilustración 24-2: Áreas en las que se aplica IoT.

Fuente: Al-Fuqaha et al., 2015, p.2347

Realizado por: Huilcamaygua, Cinthya, 2023.

Existen muchas otras más, al no estar definido su concepción está teniendo ciertos cambios conforme se desarrollan investigaciones.

2.2.4.3 Comunicación en el IoT

El Internet de las cosas (IoT) ha permitido la conexión de dispositivos inteligentes para generar, intercambiar y procesar información en tiempo real. La comunicación entre estos dispositivos y con la nube es fundamental para que el IoT funcione adecuadamente. En este sentido, las tecnologías de comunicación en el IoT son muy diversas y se adaptan a las necesidades de cada caso de uso. (Barrio, 2020, p. 26; Atzori et al., 2010)

Las comunicaciones en el IoT pueden ser inalámbricas o por cable para conectar dispositivos y enviar información a la nube. Entre las tecnologías inalámbricas más comunes se encuentra una variedad de tecnologías de comunicación, que incluyen redes de área amplia de baja potencia (LPWAN), Bluetooth, Wi-Fi, RFID y NFC. (Barrio, 2020, p. 26; Al-Fuqaha et al., 2015)

Las redes de área amplia de baja potencia (LPWAN) se han convertido en una de las redes de comunicación más populares en el IoT debido a su bajo costo y eficiencia energética. Estas redes son capaces de cubrir áreas geográficas extensas con poca potencia, lo que las hace ideales para conectar dispositivos a larga distancia y en áreas rurales como aplicaciones que requieren el monitoreo remoto de sensores, como la monitorización del clima y la calidad del aire. Ejemplos de LPWAN incluyen Sigfox, LoRaWAN y NB-IoT. (Atzori et al., 2010)

Bluetooth y Wi-Fi son tecnologías de comunicación inalámbricas comunes que se utilizan ampliamente en dispositivos y hogares inteligentes. Bluetooth es conocido por su bajo consumo de energía, lo que lo hace ideal para dispositivos que no tienen acceso constante a una fuente de energía, como los wearables[10]. Wi-Fi, por otro lado, es conocido por su alta velocidad y ancho de banda, lo que lo hace ideal para dispositivos que requieren una transferencia rápida de grandes cantidades de datos, como las cámaras de seguridad inteligentes. (Bandyopadhyay y Sen, 2011)

Radiofrecuencia de identificación (RFID) y Near Field Communication (NFC) son tecnologías de comunicación de corto alcance que se utilizan en aplicaciones como pagos móviles y control de acceso. RFID es capaz de identificar y rastrear objetos sin la necesidad de una línea de visión directa, lo que lo hace ideal para aplicaciones como el seguimiento de activos y la gestión de inventarios. NFC, por otro lado, permite la transferencia de datos entre dispositivos a través del

[10] Dispositivo electrónico que una persona puede llevar puesto

contacto cercano, lo que lo hace ideal para aplicaciones como el emparejamiento de dispositivos y los pagos móviles. (Bandyopadhyay y Sen, 2011)

Por otro lado, las comunicaciones por cable también se utilizan en el IoT, especialmente en aplicaciones industriales y de automatización. En estos casos, se utilizan protocolos de comunicación como Modbus, Profibus o Ethernet Industrial. (Botta et al., 2016)
Es importante destacar que, aunque la conectividad inalámbrica es una de las principales opciones en el IoT, su seguridad es un aspecto crítico que debe ser considerado en el diseño y desarrollo de aplicaciones de IoT. Algunas medidas de seguridad incluyen el uso de cifrado, autenticación y autorización, entre otros. (Cisco, 2020)

2.2.4.4 Categorías de IoT

El Internet de las cosas (IoT) ha dado lugar a una gran cantidad de dispositivos conectados a la red, lo que ha llevado a la creación de dos categorías principales de IoT. La primera es la Internet de las cosas industrial (IIoT), que se enfoca en la comunicación máquina a máquina (M2M) para crear redes inteligentes, ciudades inteligentes y sistemas de transporte inteligentes. En este ámbito, los dispositivos IoT se utilizan para automatizar y controlar procesos en entornos industriales, como la fabricación, el transporte, la energía y la agricultura. (Bandyopadhyay y Sen, 2011; Joyanes, 2021)

La segunda categoría de IoT es la Internet de las cosas para el consumidor (CIoT). Esta se enfoca en el uso de dispositivos IoT para mejorar el estilo de vida de las personas en el hogar y en la vida cotidiana. Los dispositivos IoT para el consumidor incluyen electrodomésticos inteligentes, sistemas de seguridad para el hogar, wearables y mucho más (Barrio, 2020).

- **IIoT**

En la investigación de *IIoT-MEC* conciben a la IoT Industrial "como una revolución que está cambiando la imagen de la industria de manera profunda" porque esta tecnología permitiría la integración hacia la red de múltiples equipos con el objetivo de controlarlos y manipularlos de manera remota o también para tomar datos, procesarlos y obtener resultados. Esta tecnología al ser industrial maneja una cantidad alta de datos por los dispositivos instalados en industrias, casas inteligentes, ciudades inteligentes, transporte entre otras. (Hou et al., 2019; Joyanes, 2021)

IIoT se implementa en ambientes de gran escala, esa es la razón por la que maneja un volumen de tráfico alto. La seguridad es crucial en este tipo de redes inteligentes, porque si se logra

deshabilitar la red o lograr que se caiga a base de ataques de denegación de servicio DDS o también con DDS distribuido puede provocar el colapso de una ciudad en el caso de una Smart city. (Joyanes, 2021)

- **CIoT**

El CIoT representa la clase de aplicaciones orientadas al consumidor en las que los dispositivos son productos de consumo, como electrodomésticos inteligentes (por ejemplo, refrigeradores, lavadoras, secadoras) y dispositivos personales (por ejemplo, sensores de estado físico, Google Glass, los sistemas de telemedicina). (Corsaro, 2014)

Las tasas de datos individuales son relativamente bajas, y los datos no están sujetos a restricciones temporales estrictas. La seguridad no juega un papel crucial en este tipo de aplicaciones y que, aunque se minimiza mucho, se requiere al menos a nivel de transporte. Finalmente, las aplicaciones CIoT apuntan a plataformas de consumo que, si bien presentan un buen nivel de heterogeneidad, en general son menos exóticas que algunas de las plataformas que se encuentran en las aplicaciones IIoT. (Corsaro, 2014; Barrio, 2020)

- Diferencias entre IIoT y CIoT

Una de las principales diferencias entre IIoT y CIoT está en la latencia producida por un dispositivo dado y los requerimientos que necesitan de la red (Corsaro, 2014).

La latencia se produce por la cantidad de tráfico que pasa por el canal y mientras más tráfico, más latencia se puede producir. IIoT produce mayor volumen de tráfico y requiere de difusión con baja latencia, mientras que los dispositivos CIoT a menudo producen velocidades de datos bajas a moderadas que tienen requisitos bajos a moderados con respecto a la latencia. (Atzori et al., 2010)

La seguridad: este factor es muy importante en ambos, pero, se le agrega prioridad a IIoT porque maneja una masiva cantidad de datos de una gran área como lo es en una ciudad, ¿qué pasaría si afectan a los semáforos de una ciudad inteligente? Pues llevaría consecuencias graves por posibles choques de automóviles. (Hou et al., 2019)

La diferencia más obvia es la escala a la que está orientada, la IIoT está orientada a una escala mucho mayor a la de CIoT porque se la aplica a ciudades, el transporte, en una industria. Y CIoT tiene una escala menor porque se la aplica en un hogar o también en el área de la salud y atención médica. (Ferney et al., 2019)

2.2.4.5 Arquitectura IoT

La arquitectura IoT se refiere al diseño estructural de los sistemas de dispositivos y su conectividad para permitir la recopilación, procesamiento y análisis de datos en tiempo real. La arquitectura de IoT de 3 capas es ampliamente aceptada a pesar de que IoT no tenga una definición universalmente aceptada. Este modelo de 3 capas fue desarrollado durante los inicios del desarrollo de IoT. A medida que ha pasado el tiempo, se han logrado avances en IoT que han dado lugar al desarrollo de nuevos modelos de arquitectura que incluyen más capas y nuevas funcionalidades. Existen varios modelos de arquitectura IoT, pero en general, todos tienen un conjunto de componentes comunes.

- **Arquitectura de 3 capas**

Esta estructura es básica desde el punto de vista técnico, cuando aún las redes IoT no han sido desplegadas o aplicadas a gran escala. (Fang et al., 2016)

El nivel de percepción o capa de dispositivos: comprende todos los medios físicos (sensores) que recolectan información del entorno (González García, 2017). A esta capa se la puede denominar los órganos de los sentidos de IoT, porque su función principal es la de identificar objetos y recopilar información. Esta capa "incluye etiquetas y lectores de códigos de barras bidimensionales, etiquetas RFID y lectores-escritores, cámara, GPS, sensores, terminales y red de sensores" (Fang et al., 2016)

El nivel de red o capa de conectividad y procesamiento: es el encargado de enlazar a los sensores y servidores entre sí para que los datos recogidos en la capa de percepción sean transmitidos y procesados. (González García, 2017). Es decir, es el cerebro de IoT ya que aquí se incluye una red de convergencia de comunicación e Internet, centro de información, centro de procesamiento inteligente, entre otras (Kaloxylos et al, 2021). El nivel de aplicación o capa de aplicación denota que aquí se puede desplegar hacia distintas áreas de aplicación como lo puede ser en el ámbito industrial (González García, 2017). (Ver Ilustración 25-2.

Ilustración 25-2: Arquitectura de 3 capas de IoT

Fuente: Al-Fuqaha et al., 2015, p.2349

Realizado por: Huilcamaygua, Cinthya, 2023.

- **Arquitectura de 5 niveles**

Esta arquitectura podría explicar de mejor manera las características de IoT. Está formada por los niveles de percepción, transporte, proceso, aplicación y nivel de negocio. En este modelo de arquitectura los niveles de percepción y aplicación son los mismos que en la arquitectura de tres niveles, como se puede ver en la Ilustración 26-2 (González García, 2017). En cuanto al resto de niveles:

El nivel de transporte transfiere los datos los datos de los sensores mediante las siguientes técnicas: redes 3G, LAN, Bluetooth, RFID y NFC, desde el nivel de percepción al nivel de proceso y viceversa (González García, 2017). Esta capa también es llamada capa de red y es la encarga de transmitir los datos recibidos de la capa de percepción al centro de procesamiento a través de varias redes, como redes inalámbricas o de cable, incluso la red de área local (LAN). En esta capa se pueden encontrar muchos protocolos de comunicación como IPv6. (Fang et al., 2016)

El nivel de proceso o capa de procesamiento: almacena, analiza y procesa grandes cantidades de datos procedentes del nivel de percepción. Las técnicas principales son bases de datos, procesamiento inteligente, computación en la nube y la computación ubicua, siendo estas dos últimas las tecnologías principalmente usadas. (Fang et al., 2016)

El nivel de negocio administra el lanzamiento y carga de las aplicaciones, el modelo de negocio y modelo de ganancias y gestiona la privacidad de los usuarios.(Fang et al., 2016)

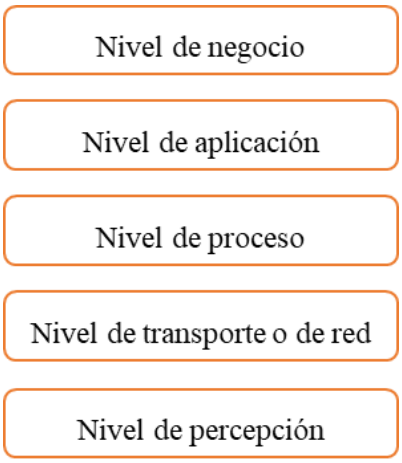


Ilustración 26-2: Arquitectura de 5 capas

Fuente: Al-Fuqaha et al., 2015, p.2350
Realizado por: Huilcamaygua, Cinthya, 2023.

2.2.4.6 Técnicas de la capa de procesamiento del modelo de 5 capas

En el modelo de 5 capas se menciona que la capa de procesamiento puede tener distintas técnicas de las cuales depende su arquitectura, las técnicas de cloud y fog computing son algunas de ellas, estas dos técnicas proporcionan a los usuarios finales, datos, cálculo, almacenamiento y servicios. (González García, 2017)

- Computación en la nube o Cloud Computing

La computación en la nube proporciona arquitectura orientada a servicios, computación utilitaria, virtualización, infraestructura como servicio, plataforma como servicio, software como servicio y otros servicios web en la nube. Los servicios de computación en la nube no necesitan inversiones de capital iniciales, pago por uso, no tienen gastos generales para administrar el centro de datos y tienen velocidad de implementación. La computación en la nube es confiable, escalable y sostenible y sobretodo es bajo demanda.
Su funcionamiento es el siguiente: envía datos a la nube procesándolos allí, manteniendo el orden de la infraestructura: los dispositivos debajo, arriba el procesamiento y por encima de ésta se encuentran las aplicaciones.(González García, 2017)

El IoT y cloud computing ha seguido una continua evolución independiente, hasta hace poco se viene integrando la tecnología IoT con la computación en la nube. Las características del Cloud Computing se vuelven atractivas para el IoT. La mayor ventaja es que Cloud Computing ofrece

recursos virtualmente sin límites a IoT que podrían compensar restricciones tecnológicas de las cuales se componen los sistemas IoT como el procesamiento. (Mell y Grance, 2011)

Un modelo de nube consta de cinco características esenciales: autoservicio bajo demanda, amplio acceso a la red, agrupación de recursos, elasticidad rápida y servicio medido. (Geng, 2017)

- Edge Computing

La computación ubicua el procesamiento es local y lo realizan los dispositivos que se encuentran al borde de la red. (González García, 2017)

Cloud Computing es el punto de partida para Edge Computing. Edge computing es un modelo basado en la concentración de datos y aplicaciones a dispositivos en el borde de la red. El procesamiento de datos locales es realizado en un dispositivo inteligente en lugar de ser enviado sin procesar mediante la red. Sus características principales son: aplicaciones y datos se localizan al borde de red con computación y almacenamiento distribuido, alta distribución geográfica con gran cantidad de nodos, datos, aplicaciones heterogeneidad de dispositivos, aplicaciones, movilidad, y comunicación en tiempo real. (Joyanes, 2021, p. 168)

Cada una de estas arquitecturas se adapta mejor a un tipo de aplicación específico. Por ejemplo, la arquitectura de borde (edge computing) es adecuada para aplicaciones en las que los datos se generan rápidamente y se necesitan respuestas en tiempo real, como en la fabricación y el monitoreo de maquinaria. La arquitectura de nube (cloud computing) es más adecuada para aplicaciones que requieren grandes cantidades de datos y análisis a largo plazo, como en la agricultura y la energía. (González García, 2017; Joyanes, 2021, p. 169)

En general, la arquitectura IoT debe ser flexible, escalable y segura para permitir el crecimiento y la adaptación a medida que las necesidades de la aplicación cambien con el tiempo. (González García, 2017)

2.2.5 Protocolos y estándares de IoT

Los fundamentos de las tecnologías de comunicación IoT están comprendidos en base a dos grandes grupos de estándares: los estándares de tecnología y los estándares regulatorios. Los estándares de tecnología comprenden a los protocolos de comunicación, los protocolos de comunicación y estándares de agregación de datos, mientras que los estándares regulatorios se

relacionan con la seguridad y privacidad de los datos. Lo que caracteriza los protocolos IoT, es que son protocolos que pertenecen a la capa de sesión del modelo OSI. Tomando como referencia al modelo OSI que consta de 7 capas, se pretende llegar al punto en que se mencione a los estándares y los protocolos de comunicación IoT, pasando por cada una de las capas. (Geng, 2017)

Cada capa del modelo OSI provee de información a la capa que le precede habiendo recibido previamente los servicios desde una capa inferior. Estas capas son la capa de aplicación, presentación, la capa de sesión, transporte, la capa de red, la capa de enlace de datos y la capa física, mencionándolos de superior a inferior. (Geng, 2017)
La capa 1 del modelo OSI es la capa física que engloba todos los medios físicos para la recopilación o envío de información. La capa de enlace de datos es la capa 2 del modelo OSI que se encarga del manejo de la información y trasladarla sin errores hacia un nodo destino, esta capa se subdivide en otras dos capas, que son el control de enlace lógico o LLC y el control de acceso a medios o MAC, ambas se encargarían de el control de flujo y el control de errores. La capa de red es la que administra las direcciones de datos, es decir, la ruta que deben seguir los datos y se asegura que los datos que envía lleguen al receptor. La capa de transporte es la responsable del intercambio de información entre dos sistemas sin errores y en orden. La capa de sesión o capa 5 es la encargada de gestionar las conexiones entre diferentes sistemas operativos y mantener esa conexión hasta su terminación, aquí se incluyen protocolos y estándares como MQTT, XMPP, DDS, AMQP, CoAP y por su puerto, los puertos por donde se comunican. La capa de presentación o capa 6 es donde se pueden aplicar seguridades a la comunicación como la encriptación de los datos. (Geng, 2017; Stallings, 2014b)

2.2.5.1 AMQP

Advanced Message Queuing Protocol o protocolo de colas de mensajes avanzado, este protocolo tiene mucho que ver con el sistema de colas en la transmisión de mensajes. Está basado en el modelo de comunicación publicación/suscripción en donde un dispositivo central o middleware recoge los mensajes a manera de colas. Está orientado a la no perdida de mensajes con el uso del protocolo TCP, por eso es utilizado para enviar mensajes transaccionales entre servidores denominado servidor a servidor (S2S). Surgió de la necesidad de intercambiar información entre entidades bancarias donde el término confiabilidad se destacaba.(Geng, 2017; Jeyaraman et al., 2012)

En el contexto de IoT, AMQP es más apropiado para el plano de control o las funciones de análisis basadas en servidor. (Geng, 2017)

2.2.5.2 XMPP

El protocolo de extensible de mensajería y presencia o XMPP es un estándar de mensajería para conectar a las personas mediante mensajes de texto. Su nombre indica su uso, presencia, especificando claramente que las personas o dispositivos deben estar presentes. Se considera que es uno de los mejores protocolos de comunicación para conectar unos dispositivos con otros en un entorno de transacciones de dispositivo a servidor (D2S) tal como MQTT. Su arquitectura es centralizada y su modelo de comunicación es publicación/suscripción. (Geng, 2017; Al-Fuqaha et al., 2015)

- Es un protocolo de código abierto.
- Este protocolo se comunica con ayuda del protocolo TCP en la capa de transporte.
- Utiliza el formato de texto XML de forma nativa, que facilita la comunicación.
- Sobre XMPP se puede utilizar el protocolo HTTP en la parte superior.
- Es un protocolo escalable
- El direccionamiento de cada dispositivo hacia el servidor es nombre@dominio.com

2.2.5.3 DDS

Sus siglas en español significan Servicio de Distribución de Datos, es un estándar desarrollado por la OMG u Object Management Group dedicado a transmitir información en sistemas que necesitan datos en tiempo real. DDS es un sistema abierto que utiliza el modelo de comunicación publicación/suscripción dirigida para el intercambio de datos en tiempo real entre dispositivo y servidor (D2S). Fue creado con la intención de proporcionar un envío rápido y seguro a través de redes o entornos donde intervengan dispositivos IoT (Object Management Group, 2021).

DDS no es un protocolo en sí, es un estándar que proporciona todas las facilidades para que todos los dispositivos conectados se comuniquen sin demoras en la entrega de los mensajes y de manera segura y esté disponible todo el tiempo, pues integra los componentes de un sistema operativo y a la vez es un lenguaje de comunicación. (Corsaro, 2014; González García, 2017)

La razón de porque actúa como lenguaje y como sistema operativo es porque DDS se ubica en la capa de software o middleware entre la de aplicación y la de plataforma de un sistema distribuido (Ver Ilustración 27-2), administrando todas las acciones que ocurren debajo del middleware (los detalles de bajo nivel) como el formato del cable de datos, el descubrimiento, las conexiones, la confiabilidad, los protocolos, la selección de transporte, la QoS, la seguridad, etc. Como se resultado se efectúa un intercambio de datos más sencillo y simplificando el desarrollo de sistemas

distribuidos al permitir que los desarrolladores de software se centren en el específicamente de sus aplicaciones en lugar del proceso de pasar información entre aplicaciones y sistemas. (González García, 2017; Object Management Group, 2021)

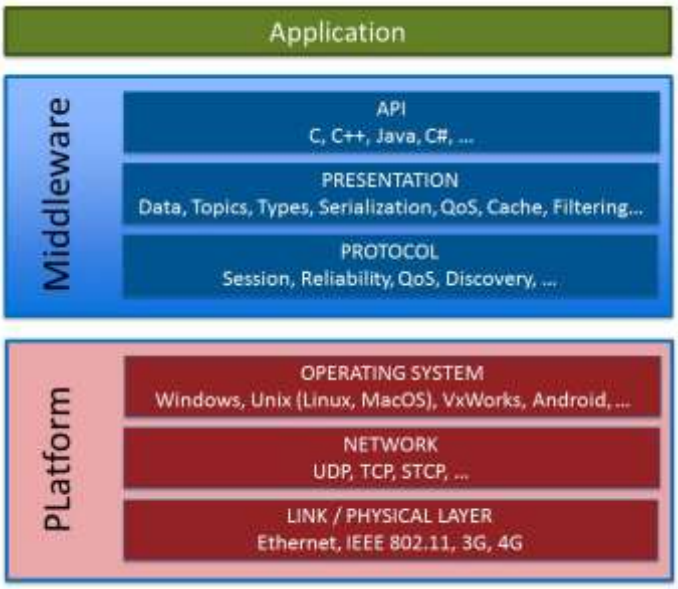


Ilustración 27-2: Capas de un sistema distribuido basado en el modelo OSI de 7 capas.

Fuente: https://www.dds-foundation.org/what-is-dds-3/

Realizado por: Huilcamaygua, C. 2022

2.2.5.4 MQTT

Message Queuing Telemetry Transport o MQTT es un protocolo que ofrece una comunicación ligera, de configuración simple y fácil de implementar, características que son de mucha ayuda pues la vuelven ideal para la comunicación IoT en ambientes de comunicación máquina a servidor (M2S). Este protocolo, como lo dice su propio nombre, su propósito es la telemetría o monitoreo remoto, la idea es recoger datos o información de distintos dispositivos y transportarlos a través de la red y monitorear esos datos desde la nube. (Geng, 2017, p. 155; OASIS, 2019, p. 1)

El protocolo destaca por su enfoque de comunicación publicación/suscripción, funcionando sobre TCP y permitiendo también la utilización de HTTP/Json. Presenta cabeceras mínimas para ser un protocolo de peso ligero y ofrece un transporte independiente del contenido de la carga útil. Proporciona opciones de QoS configurables para garantizar la entrega del mensaje, y su middleware es eficiente en el uso de recursos. Su funcionamiento se basa en mensajes con tópicos y contenido. El middleware es un tema importante, pues es que el gestiona la recepción y la entrega de los mensajes, se lo puede llamar intermediario o más técnicamente bróker. La función

del bróker es la de intercambiar información por medio de la publicación y suscripción de los datos. (OASIS, 2019, p. 1)

- EMQ X Bróker

EMQ es una solución practica al momento de transmitir datos de forma rápida, fiable y segura en todo tipo de escenarios como: escenarios de prueba para investigaciones, un proyecto de IoT a gran escala, proyectos escolares, proyectos de control y automatización, escenarios de tipo doméstico con pocos dispositivos conectados o simplemente para monitorear todos los dispositivos IoT que quieras conectar porque abarca tantos usuarios se desee con un límite de 10000. Además, este bróker se destaca por su facilidad de uso ya que se maneja a través de una interfaz web y su página oficial cuenta con toda la información necesaria para su uso con documentación acerca de todas las funciones que tiene. (EMQ, 2022)

EMQ X es un bróker de código abierto que ofrece protocolos MQTT 3.1/5.0, clustering sin maestro, alta disponibilidad y baja latencia, siendo altamente escalable hasta 10000 dispositivos conectados. La versión Enterprise admite hasta un millón de dispositivos, múltiples protocolos IoT, un potente motor de reglas SQL y diversas bases de datos de persistencia. Además, brinda integración flexible con bases de datos y Kafka, junto con un Centro de Gestión y Monitoreo. La plataforma EMQ X alcanza 10 millones de conexiones simultáneas, alta fiabilidad, soporte 5G y una integración ágil con sistemas heredados, haciendo hincapié en la interoperabilidad y escalabilidad en entornos de nube e IDC cruzada. (EMQ, 2022)

2.2.6 Realidad virtual

2.2.6.1 Introducción a la realidad virtual

El término realidad virtual está trascendiendo a pasos agigantados en los últimos años alrededor del mundo en el campo de la tecnología e innovación, y que podría llegar a convertirse en la tecnología de las nuevas generaciones, cambiando así la vida cotidiana de muchos y abriendo puertas a nuevos proyectos puesto que su objetivo es el de engañar a los sentidos del cuerpo humano, tal como es en el campo de la educación donde los estudiantes tendrán a su disposición todas las herramientas virtualizadas y empezar a experimentar en un laboratorio virtual sin el peligro de que ocurra algún accidente como sucede en el ambiente real. (Simiscuka et al., 2019). Pero no solo funciona para el campo de la educación, esto se extiende a distintas áreas o campos como lo son de la ingeniería, medicina, mecánica, artes, e inclusive deportes. (Micheng y Liping, 2020; Lan et al., 2021)

2.2.6.2 Definición

En el paper *Virtual Reality: A brief survey* la realidad virtual es denominada como:

> *La realidad virtual se basa en la noción de inmersión, es decir, un nuevo avance tecnológico en el campo de la interacción hombre-máquina que lo acerca a la vida real. Es una tecnología para la simulación de un mundo real o virtual en el que uno puede sumergir, tocar y sentir los objetos con presencia virtual en ese mundo tridimensional.*(Singh y Singh, 2017)

La idea de realidad virtual se la puede definir de diversas maneras, todas en contexto tienen el mismo objetivo. "La realidad virtual se puede definir como una forma de simular el mundo real mediante la aplicación de la teoría de inmersión en un entorno 3D virtual" (Simiscuka et al., 2019)

Para entrar en ese ambiente simulado se estimulan los sentidos del ser humano que principalmente son la visión, el oído y el tacto.

La realidad virtual una representación del mundo o entorno real simulado a computadora que, a través de dispositivos, sumerge a un usuario haciéndole creer que está dentro de ese ambiente simulado.

2.2.6.3 Motores de realidad virtual y creación de entornos virtuales

- Unity 3D

Unity 3D es una plataforma de videojuegos que ayuda a crear figuras y ambientes tanto en 2D como en 3D de forma interactiva. Es un software utilizado por artistas, diseñadores y desarrolladores para elaborar distintas figuras y entornos inmersivos para ofrecer la mejor experiencia interactiva, con la facilidad de ejecutarlo en distintos dispositivos de distintas plataformas. (Unity, 2022)

Es una de las plataformas más utilizadas y amadas por los programadores y creadores de videojuegos por su facilidad de manejo, amplia documentación y una comunidad que a diario se apoya en los foros. En la Ilustración 28-2 está el logo de Unity 3D.

Ilustración 28-2: Logo Unity 3D

Realizado por: Huilcamaygua, Cinthya, 2023.

Fuente: https://unity3d.com/profiles/unity3d/themes/unity/images/pages/branding_trademarks/unity-masterbrand-black.png

- Unreal Engine

Este motor de juegos es una herramienta de código abierto para la creación de ambientes en 3D y ofrece una calidad visual excelente para entornos inmersivos. También brinda una amplia documentación, y la comunidad está siempre activa. Este tipo de herramienta esta más enfocado a dar texturas y más dimensión a los personajes o paisajes que se creen dentro de él, especial para construir un videojuego, o inclusive para simular el movimiento fluido si se decidiera utilizar para presentar una película netamente en 3D. (Epic Games Inc, 2022)

Ilustración 29-2: Logo de Unreal Engine.

Fuente: https://cdn2.unrealengine.com/ue5-logo-stacked-unreal-engine-white-677x545-ad05f25fd314.png

2.2.7 HTTP

2.2.7.1 Definición

El protocolo HTTP o por sus siglas HiperText Transfer Protocol, es el protocolo que está detrás de toda la comunicación por la WWW. Este protocolo pertenece al modelo de comunicación

Cliente – Servidor y es usado globalmente para transmitir información entre un usuario y un servidor.

La RFC2616 menciona a HTTP de la siguiente manera:

> *El Protocolo de transferencia de hipertexto (HTTP) es un protocolo de nivel de aplicación para sistemas de información hipermedia distribuidos y colaborativos. Es un protocolo genérico, sin estado, que se puede utilizar para muchas tareas más allá de su uso para hipertexto, como servidores de nombres y sistemas de gestión de objetos distribuidos, a través de la extensión de sus métodos de solicitud, códigos de error y encabezados […]. Una característica de HTTP es la escritura y negociación de la representación de datos, lo que permite que los sistemas se construyan independientemente de los datos que se transfieren.* (Fielding et al., 1999)

El protocolo HTTP es un protocolo que funciona a nivel de la capa de aplicación y transmite la información con cierta ligereza y velocidad. Sigue el modelo de conexión TCP/IP. Los componentes de esta comunicación con HTTP serian un equipo cliente o host y un equipo servidor. El host realiza peticiones o solicitudes hacia el equipo servidor a través de una URL, el servidor siempre está a la espera de solicitudes de conexión de uno o varios equipos host. De esta forma, se concluye que la forma de operación de este protocolo es de solicitud y respuesta. (Fielding et al., 1999)

El tipo de conexión en la cual se basa para la comunicación es la de TCP/IP. Este protocolo utiliza el protocolo de transporte TCP en el número de puerto 80 por defecto, que trataría de mantener la conexión en la comunicación y además garantiza la entrega de mensajes entre los equipos involucrados. (Fielding et al., 1999)

2.2.7.2 Tipos de mensaje

- HTTP Request

Es todo mensaje que origina el cliente solicitando un recurso al servidor.(Fielding et al., 1999)

En este tipo de mensaje se utiliza una dirección URL y sus componentes para completar la información necesaria para realizar la solicitud. Esta dirección URL proporciona los siguientes elementos: el nombre del host o anfitrión, el número de puerto, la ruta del recurso en el servidor y una cadena de consulta (como una búsqueda por filtro). (IBM Corporation, 2015a)

El HTTP Request utiliza algunos campos de la URL para completar sus elementos, que son: la línea de solicitud (o Request Line), cabeceras HTTP y el cuerpo del mensaje.

La línea de solicitud es la primera línea en estar presente en el HTTP Request. Contiene el método a usar en la comunicación, la ruta de acceso y la versión de HTTP, y en algunos casos una cadena de consulta (IBM Corporation, 2015d). Todos estos elementos son importantes, pero del método depende el tipo de petición que se le hace al servidor.

Las cabeceras HTTP o campo de cabeceras son datos que contienen información útil para los dispositivos que intervienen en la comunicación, y le hacen saber destinatario cual es el remitente y como quiere comunicarse este remitente.

El cuerpo del mensaje es el mensaje con la información real y útil para el destinatario, generalmente el cuerpo del mensaje es utilizado con ayuda del método POST para publicar o postear información en un recurso de un servidor.

- HTTP Response

HTTP response es un mensaje respuesta por parte de servidor hacia el cliente que indica que se ha procesado su solicitud, sea para proporcionar al cliente el recurso que solicitó o para informarle que ha existido un error. Cualquiera sea el caso, el servidor responde con este mensaje de response. (IBM Corporation, 2015d)

Este mensaje contiene los campos: Línea de estado o status line, cabeceras HTTP y el cuerpo del mensaje
La línea de estado es la primera en estar presente en el mensaje de respuesta, esta contiene 3 elementos necesarios para el intercambio de datos que son: la versión de HTTP con la cual responde el servidor; código de estado que es un grupo de 3 dígitos que según el valor que indica el estado de la solicitud y el texto de estado, que va de la mano con el código de estado y no es más que un mensaje legible para el ser humano del estado de la solicitud. (IBM Corporation, 2015d; Fielding et al., 1999)

Las cabeceras HTTP se compone de información que el cliente acerca del mensaje de respuesta, son datos útiles que pueden ser guardados en la cache y usados posteriormente.

El cuerpo del mensaje puede contener información acerca del estado de la respuesta. Si se refiere a la respuesta de una solicitud procesada correctamente, el cuerpo del mensaje contendrá datos

del recurso solicitado, mientras que, si fuera una respuesta de una solicitud errónea, el cuerpo del mensaje contendrá información de acerca de las razones del fallo. (IBM Corporation, 2015d; Fielding et al., 1999)

2.2.7.3 Métodos y códigos de estado

- Métodos

Los métodos también llamado token de método, es un comando que le dice al servidor que es lo que tiene que hacer con el recurso. Es un campo que distingue entre mayúsculas y minúsculas.

Los métodos comúnmente utilizados son: GET, POST, HEAD que son utilizados en los servidores de propósito general. Estos tres comandos representan la ejecución de las acciones obtener información, publicar información y chequeo de estados de las cabeceras. También existen otros métodos como el OPTIONAL, PUT, DELETE, TRACE y CONNECT. Existen otros métodos de extensión con tokens para otros propósitos. (Fielding et al., 1999)

El método GET, existente desde la primera versión de HTTP/1.0, es utilizado para recibir o recoger información solicitada al servidor. Un ejemplo claro de este método es cuando se da clic sobre un enlace de descarga en una página web que como resultado tenemos al servidor HTTP que responde a esa solicitud de la URL retornando una respuesta con la información pedida. También se halla el método GET condicional y ocurre cuando en el campo de la cabecera del request se incluye un If-Modified-Since. Este GET condicional cumple una función dentro de la comunicación con el servidor pues trata de reducir el uso de la red y actualiza la cache sin necesidad de enviar varias solicitudes. (Fielding et al., 1999)

El método HEAD es un método muy similar al método GET ya que le solicita al servidor le proporcione información, solo que en este método HEAD no retorna información que deba ser de utilidad para el usuario como sucede con el GET, solo que son de las cabeceras. Generalmente este comando pide información sobre un objeto como tamaño, fecha de modificación. Este tipo de método es muy útil al momento de comprobar la accesibilidad y disponibilidad de los enlaces de hipertexto con el fin de conocer cuando se debe realizar una actualización de los ficheros guardados en la caché. (Fielding et al., 1999)

El método POST se ocupa para almacenar información en el recurso de un servidor. Dicho en otras palabras, solicita al servidor que acepte la información incluida en la solicitud como un

nuevo dato. Esta maniobra o acción se debe realizar hacia una URL destino que permita el guardado. Se la utiliza generalmente para guardar datos ingresados en un formulario. (Fielding et al., 1999)

De manera resumida, GET es utilizado para realizar consultas en una URL transmitiéndose a través de una cadena de consulta, mientras que POST incluye los datos que quiere subir al servidor en el cuerpo del mensaje

- Códigos de estado y texto de estado

Los códigos de estado son valores compuestos por 3 dígitos, lo cual proporciona una gran cantidad de códigos. El texto de estado o reason phrase es un mensaje que resume el código del estado haciéndolo legible para el ojo humano. Los códigos de estado HTTP se dividen en diferentes rangos para proporcionar información sobre la respuesta de una solicitud. (Fielding et al., 1999)

El rango de códigos 100-199 se clasifica como informativo y generalmente no tiene un impacto significativo en la respuesta. El rango de códigos 200-299 indica una respuesta satisfactoria o correcta, lo que significa que la solicitud se ha procesado correctamente. El rango de códigos 300-399 indica que se ha realizado una redirección. Esto significa que el servidor está indicando al cliente que realice una acción adicional para completar la solicitud. El rango de códigos 400-499 indica errores por parte del cliente. Estos códigos se generan cuando la solicitud del cliente es incorrecta o no se puede cumplir. El rango de códigos 500-599 indica errores del servidor. Estos códigos se generan cuando el servidor encuentra un problema al procesar la solicitud del cliente. Códigos superiores a 600 no están definidos en la especificación del protocolo HTTP y no tienen un significado específico. (Fielding et al., 1999)

Estos rangos de códigos de estado HTTP proporcionan una forma estandarizada de comunicar el resultado de una solicitud y ayudan a identificar y solucionar problemas en la comunicación entre el cliente y el servidor. (Fielding et al., 1999)

En la tabla 1-I, se encuentran algunos de estos códigos de estado con su texto, pero los códigos más comunes de encontrar son el 200, 201, 204, 400, 404, 503, 505. Aquí también existen las extensiones de protocolos, que es muy poco común ocuparlos. (Fielding et al., 1999; IBM Corporation, 2015c)

Tabla 2-1: Códigos de estado y texto de estado que pueden presentarse en el mensaje HTTP response.

Código de estado	Texto de estado	Código de estado	Texto de estado	Código de estado	Texto de estado	Código de estado	Texto de estado
100	Continuar	301	Movido permanentemente	404	No encontrado	414	Solicitud-URI demasiado grande
101	Protocolos de conmutación	302	Encontrado	405	Método no permitido	415	Tipo de medio no compatible
200	OK	303	Ver otros	406	No aceptable	416	Rango solicitado no satisfactorio
201	Creado	304	No modificada	407	Autenticación de proxy requerido	417	Expectativa fallida
202	Aceptado	305	Usar proxy	408	Solicitud de tiempo de espera	500	Error interno del servidor
203	Información no autorizada	307	Redirección temporal	409	Conflicto	501	No implementado
204	Sin contenido	400	Solicitud incorrecta	410	Desaparecido	502	Puerta de enlace incorrecta
205	Restablecer contenido	401	No autorizado	411	Longitud requerida	503	Servicio no disponible
206	Contenido parcial	402	Pago requerido	412	Precondición fallida	504	Tiempo de espera de puerta de enlace
300	Opciones múltiples	403	Prohibido	413	Entidad de solicitud demasiado grande	505	Versión HTTP no compatible

Fuente: Fielding et al., 1999, p.27

Realizado por: Huilcamaygua, Cinthya, 2023.

Cabe destacar que no todos los códigos de estado requieren un cuerpo de respuesta o mensaje de respuesta puesto que no son admitidos, estos son los códigos 204, 205 y 304. (IBM Corporation, 2015c)

2.2.7.4 Funcionamiento

Se conoce que se efectúa la comunicación cuando el cliente realiza una solicitud hacia el servidor y este le ofrece una respuesta según lo que haya pedido. Pero en este proceso intervienen otros elementos que son de suma importancia conocer para comprender de mejor manera como se realiza el intercambio de los datos. (Fielding et al., 1999)

Cuando se realiza la petición, el cliente de manera inconsciente también ejecuta otras operaciones o acciones para realizar dicha solicitud:

En primer lugar, el usuario coloca la URL en la barra de búsqueda de un navegador web. Luego, este navegador o cliente web desmenuza a la URL separándola en distintas secciones que puede identificar como: el protocolo de acceso, la dirección IP del servidor, el puerto por donde se transmite. Después de esto, se establece un enlace o una conexión con el servidor al puerto TCP (generalmente al puerto 80). Enseguida se realiza la ejecución de la solicitud con información

como el tipo de método (GET, POST, etc), la versión del protocolo HTTP, la dirección URL de servidor, e información opcional para el servidor. Finalmente, el servidor le devuelve una respuesta al usuario, y al hacerlo se cierra la conexión entre el usuario y el servidor. Cada vez que el usuario quiera conectarse al servidor, va a realizar estas acciones. (Fielding et al., 1999)

CAPÍTULO III

3. MARCO METODOLÓGICO

3.1 Enfoque de Investigación

El enfoque de investigación seleccionado para este estudio es de carácter mixto, lo que implica la combinación de elementos cualitativos y cuantitativos en la recopilación y análisis de datos (Hernández-Sampieri y Mendoza, 2018). Esta elección se fundamenta en la necesidad de comprender tanto los aspectos teóricos y conceptuales relacionados con los protocolos de comunicación IoT, como en la evaluación empírica de su desempeño en escenarios reales de acceso a servicios de realidad virtual. Aunque el componente cuantitativo es prominente, con medidas como el tiempo de ida y vuelta, pérdida de paquetes y consumo de ancho de banda, también hay un componente cualitativo significativo.

En esta investigación, se utiliza un enfoque cuantitativo porque existe interés en conocer el rendimiento de un protocolo de IoT en redes de acceso a servicios de realidad virtual. Se realizarán mediciones y pruebas de rendimiento para obtener datos objetivos y cuantificables relacionados con parámetros técnicos de rendimiento. Estas mediciones permitirán evaluar el desempeño de cada protocolo en situaciones prácticas y verificar si el protocolo IoT seleccionado para el estudio, es más eficiente en comparación con HTTP en el acceso a servicios de realidad virtual.

El componente cualitativo reside en la revisión detallada de la literatura especializada sobre protocolos e infraestructura IoT. Esta revisión bibliográfica permitirá obtener una comprensión profunda de los protocolos de comunicación utilizados en entornos IoT y su aplicabilidad en la implementación de redes para acceso a servicios de realidad virtual. Además, se llevará a cabo un análisis comparativo de los protocolos IoT con otros protocolos de comunicación tradicionales como HTTP, con el objetivo de identificar sus ventajas y desventajas en el contexto específico de las redes de acceso a servicios de realidad virtual.

Asimismo, esta revisión teórica es la que ayuda a elaborar el diseño de la red como la selección del intermediario de la comunicación, el motor de realidad virtual, la base de datos, entre otros aspectos. Este proceso de selección, aunque se basa en la evidencia documental, también requiere interpretación y juicio para evaluar la relevancia y la aplicabilidad de la literatura a este estudio en particular.

3.2 Nivel de Investigación

Para el desarrollo de esta investigación, se empleará un nivel de investigación descriptivo y exploratorio, ya que se busca obtener una comprensión profunda del problema de acceso a servicios de realidad virtual en redes orientadas al IoT, así como describir y analizar en detalle los protocolos de comunicación a utilizar en dichas redes.

Los resultados obtenidos a través del enfoque descriptivo permitirán fundamentar y contextualizar los hallazgos del enfoque exploratorio, lo que fortalecerá la validez y relevancia de los resultados finales. (Hernández-Sampieri y Mendoza, 2018)

3.2.1 Nivel de investigación descriptivo

El nivel de investigación descriptivo permitirá caracterizar y describir de manera precisa los protocolos de comunicación más utilizados en entornos IoT, así como identificar sus principales características y funcionalidades. Se llevará a cabo una revisión exhaustiva de la literatura especializada, documentos técnicos y estándares relevantes en el área de IoT y comunicación de datos, lo que permitirá obtener una visión clara y detallada de los protocolos existentes, sus aplicaciones y posibles limitaciones. (Hernández-Sampieri y Mendoza, 2018)

Además, mediante el enfoque descriptivo, se podrán recopilar datos sobre las ventajas y desventajas de cada protocolo en el contexto específico de las redes de acceso a servicios de realidad virtual. Es decir, se hará el análisis de los protocolos y su integración con realidad virtual, proporcionando una base sólida para la posterior comparación con otros protocolos, como HTTP.

3.2.2 Nivel de investigación exploratorio

Se presenta un enfoque exploratorio en la medida en que se introduce el uso de un protocolo IoT en un nuevo contexto, que es la realidad virtual. Este aspecto se apoya en la revisión de la literatura y la selección teórica del protocolo más adecuado para este propósito. En este sentido, el estudio apunta a desentrañar cómo el protocolo IoT elegido podría ser utilizado eficientemente en un contexto de realidad virtual, una aplicación novedosa para este protocolo. Este nivel de investigación implicará la realización de pruebas piloto con el protocolo IoT seleccionado, experimentos y análisis empíricos para obtener información cualitativa y cuantitativa que complemente la revisión teórica (Hernández-Sampieri y Mendoza, 2018). Si bien existen investigaciones previas sobre ambas tecnologías por separado, la integración de ambas en este contexto específico es un campo emergente y en constante evolución.

3.2.3 Nivel de investigación correlacional

Este estudio también tiene un carácter correlacional. Una vez que se implementa el sistema de realidad virtual con el protocolo IoT elegido, se realiza una comparación directa con un sistema similar implementado con el protocolo HTTP, ampliamente usado y documentado. El rendimiento de ambos sistemas se mide y se compara para determinar si el uso de IoT ofrece ventajas significativas en términos de eficiencia y rendimiento. Este aspecto del estudio busca establecer una correlación entre la elección del protocolo de comunicación y el desempeño del sistema en un entorno de realidad virtual

3.3 Diseño de investigación

El diseño de investigación cuasiexperimental será el más indicado para el presente trabajo. Si bien este diseño no permite una manipulación directa de la variable independiente, se realizarán ajustes y configuraciones específicas en cada escenario para evaluar el rendimiento y eficiencia de los protocolos en un ambiente controlado. (Hernández-Sampieri y Mendoza, 2018)

3.3.1 Según la manipulación o no de la variable independiente

Dentro del diseño cuasiexperimental, se abordará la manipulación de la variable independiente de manera indirecta. En este caso, la variable independiente se refiere al protocolo de comunicación utilizado en cada escenario, es decir, HTTP y el de IoT (MQTT, CoAP, XMPP, etc). Si bien no se podrá intervenir directamente en la naturaleza de estos protocolos, sí se realizarán ajustes en la configuración de los escenarios experimentales para asegurar que todas las demás variables sean controladas y comparables.

A través de esta manipulación de la variable independiente, se buscará determinar si existen diferencias significativas en el rendimiento y eficiencia de la red y el servidor, lo que permitirá responder a la pregunta de investigación sobre la superioridad relativa de MQTT frente a HTTP en el acceso a servicios de realidad virtual orientado al IoT.

- Variable independiente: Protocolos HTTP e IoT elegido.
- Variables dependientes: Latencia, consumo de ancho de banda, perdida de paquetes.

3.3.2 Según las intervenciones en el trabajo de campo

En cuanto a las intervenciones en el trabajo de campo, se optará por un enfoque transversal. El estudio se llevará a cabo en un período de tiempo determinado, y se realizarán las mediciones y pruebas necesarias en ambos escenarios de manera simultánea. Ambas redes se construyen y evalúan durante el mismo periodo de tiempo.

3.4 Tipo de estudio (documental/de campo)

En esta investigación se adoptará un enfoque de estudio mixto, combinando elementos documentales y de campo, con el fin de obtener una perspectiva completa y sólida sobre el acceso a servicios de realidad virtual en redes orientadas al IoT.

3.4.1 Estudio Documental

En primer lugar, se llevará a cabo un estudio de tipo documental para recopilar información teórica y conceptual sobre los protocolos de comunicación más utilizados en entornos IoT y su aplicación en redes de acceso a servicios de realidad virtual. Esta fase implicará la revisión sistemática y crítica de la literatura especializada, artículos científicos, estándares técnicos y documentos relevantes en el área de IoT y comunicación de datos. La investigación documental permitirá establecer una base sólida de conocimiento, identificar tendencias, ventajas y desventajas de los protocolos de internet de las cosas, y comprender su relevancia en el contexto de la realidad virtual. Además, la fase documental incluirá el análisis comparativo de los protocolos seleccionados con otros protocolos de comunicación, destacando las diferencias clave y los escenarios óptimos de implementación para cada uno.

3.4.2 Estudio de campo

Tras la revisión documental, se procederá con un enfoque de estudio de campo para la implementación y evaluación práctica de los dos escenarios experimentales: uno con el protocolo IoT y otro con HTTP. En esta fase, se llevarán a cabo pruebas y mediciones reales en ambientes controlados que simulan situaciones reales de acceso a servicios de realidad virtual. Se recopilarán datos empíricos sobre el rendimiento de los protocolos en términos de latencia, consumo de ancho de banda, pérdida de paquetes y otros parámetros relevantes.

El enfoque de campo permitirá evaluar el desempeño de los protocolos en condiciones más cercanas a las aplicaciones prácticas, lo que brindará resultados más aplicables y útiles para la toma de decisiones en futuras implementaciones de redes de acceso a servicios de realidad virtual en entornos IoT.

3.5 Población y Planificación, selección y cálculo del tamaño de la muestra

El presente trabajo tiene varios componentes en su estructura. Es decir, el escenario de red debe conformarse por varios elementos que deben ser seleccionados de tal manera que cumplan con todos los objetivos planteados. Al ser un trabajo de investigación de tipo exploratorio hay que elegir cada uno de estos componentes en función del protocolo de comunicación (IoT y HTTP) y los requerimientos de la red, de manera que van a existir distintos tipos de poblaciones según el componente de la red. Asimismo, uno de los componentes principales para arrancar la investigación seria la selección de las redes de acceso a servicios de RV.

3.5.1 Planificación

Antes de continuar con el diseño de la red, se deben establecer algunos requerimientos con la finalidad de tener un sistema con una estructura definida. Este trabajo está enfocado para ser utilizado en un ambiente institucional, esto fue determinado gracias a sugerencias dadas por el tutor y asesor del trabajo de investigación. Por esta razón se requieren ciertos criterios adicionales antes de elegir los componentes de la red. Estos son:

- Crear la aplicación de realidad virtual de tal manera que puedan interactuar dos estudiantes dentro de un laboratorio 3d, estos estudiantes son los responsables de generar información a través de un brazo robótico que será almacenada y visualizada por el profesor. El profesor tendría acceso a los datos guardados y visualizarlos en el laboratorio y también podría ver los movimientos generados por el estudiante en tiempo real mientras se encuentren conectados al mismo tiempo.

- Al ser una red que debe orientarse a un ambiente institucional, los usuarios estudiantes deberían tener la capacidad de conectarse a través de internet para realizar la práctica en el laboratorio 3d.

- Todos los componentes de la red deben ser de código abierto y gratuitos.

- En este proyecto de investigación, es imperativo diseñar una red que sea coherente y compatible con la infraestructura existente de la red institucional. Esto es crucial, ya que el objetivo final es implementar nuestra solución dentro de esta red preexistente

Es importante destacar que en la elección de todos los componentes de la red se ha realizado la toma de una muestra no probabilística de cada uno de estos componentes, según las características y necesidades de la red. El método de porcentaje ponderado permite tomar decisiones cualitativas sobre diversas opciones en base a criterios establecidos y que son relevantes para este estudio. Por esta razón este método es el que más se ajusta para determinar las muestras del presente trabajo de investigación.

La ilustración 1-3 muestra cómo se determinarían los aspectos clave de la red. Se sigue un proceso de diseño de red que considera los requisitos y objetivos específicos establecidos. A partir de esta información, se puede seleccionar un protocolo de comunicación IoT que cumpla con los demás requisitos de la red para el desarrollo y diseño del escenario con el protocolo IoT para el acceso a servicios de realidad virtual.

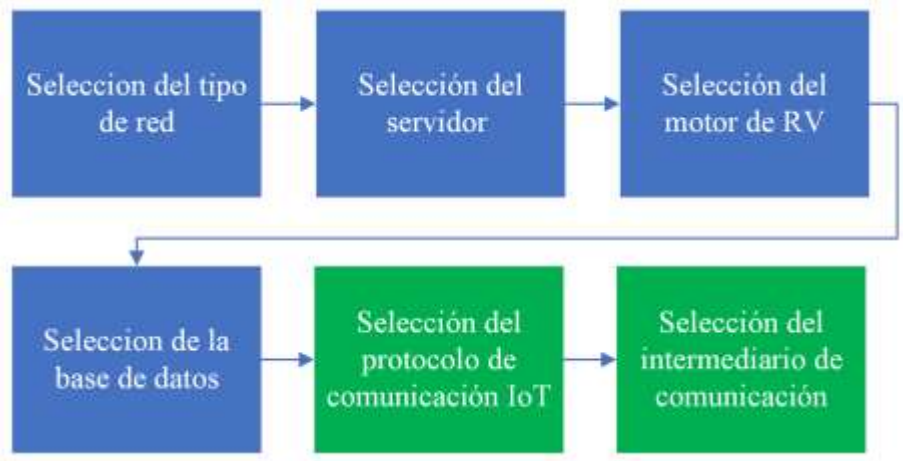


Ilustración 1-3: Diagrama de procesos para el desarrollo y diseño del escenario con el protocolo IoT

Realizado por: Huilcamaygua, Cinthya, 2023.

En el diagrama de procesos, hay dos pasos resaltados en verde. Estos pasos representan decisiones críticas que el investigador debe analizar detenidamente puesto que corresponde al cumplimiento de uno de los objetivos específicos del trabajo de titulación.

3.5.2 Método para la selección de elementos

En el marco de esta investigación, la metodología de elección de muestra adoptada es la muestra no probabilística. En este enfoque, la selección de elementos no se rige por la asignación de probabilidades, sino que se basa en criterios determinados por el investigador, lo cual guía la elección de los elementos a ser incluidos en la muestra. (Hernández-Sampieri y Mendoza, 2018)

Para una elección más precisa, el método más adecuado es la evaluación ponderada. A pesar de que no es un método de la muestra no probabilística, la evaluación ponderada es una técnica utilizada en la toma de decisiones para evaluar y comparar opciones diferentes teniendo en cuenta varios criterios. El método ponderado permite tener en cuenta de manera sistemática y objetiva múltiples factores en la toma de decisiones. Útil cuando se enfrentan opciones complejas con diferentes aspectos a considerar y se busca encontrar la mejor solución en función de las prioridades establecidas. (Marketing Publishing, 2012, p. 196)

En este método, cada criterio se asigna un peso relativo que refleja su importancia en la decisión final. Estos pesos son valores numéricos que indican cuánto impacto tiene cada criterio en la elección, y la suma de todos los criterios debe dar un total del 100%. Mientras que a cada opción se le asigna una calificación de 0 al 5 considerando como 5 a la mejor opción y 1 a la peor opción. (Marketing Publishing, 2012, p. 196)

De manera que, para cada opción hay que asignarle una calificación por criterio a evaluar, y se utiliza la siguiente fórmula para tener el valor o puntaje ponderado.

$$Evaluación\ Ponderada = peso\ del\ criterio\ \% * calificación$$

3.5.3 Selección de la red de acceso a servicios de realidad virtual

- Población

Dado que este estudio busca diseñar e implementar una red para acceso a servicios de RV orientado a IoT, la población de estudio incluye diferentes redes que se pueden usar para implementar dichos servicios. En la Tabla 1-3 se pueden observar las categorías de redes para ser usados dentro del contexto de investigación con sus respectivas justificaciones.

Tabla 1-3: Características de la población de las redes de acceso a servicios de realidad virtual.

Categorías de redes	Características
Redes Edge Computing	Este tipo de red procesa los datos cerca del borde de la red, donde se generan los datos. En el contexto de la RV, esto podría significar procesar los datos de RV en el dispositivo del usuario o en un servidor cercano, en lugar de enviar los datos a un servidor remoto para su procesamiento. Puede ser costoso implementarlo
Redes Cloud Computing	Estas redes utilizan servidores en la nube para almacenar, procesar y transmitir datos, lo que permite a los usuarios acceder a servicios de RV a través de Internet. Se otorga recursos de procesamiento a demanda con un costo adicional.

Redes privadas virtuales (VPN)	Una VPN permite a los usuarios acceder a los recursos de una red privada a través de Internet. En el contexto de la RV, esto podría permitir a los usuarios acceder a los servicios de RV de una organización de forma segura desde cualquier lugar.
Red centrada en el servidor	Estas redes mantienen todos los datos y las aplicaciones en un servidor central. Los usuarios acceden a estos recursos a través de dispositivos clientes, que pueden estar conectados a la red de la organización o a través de internet. No confundir con la red de área local.

Realizado por: Huilcamaygua, Cinthya, 2023

- Selección de la muestra no probabilística

Esta elección se basa en los criterios o requerimientos necesarios para el diseño de la red. En este caso en particular, el criterio más importante seria la compatibilidad con la red institucional existente. La ESPOCH cuenta con una sala de servidores dedicada, en donde el servidor principal se encuentra físicamente dentro de las instalaciones de la institución y cada usuario puede acceder a sus servicios a través de internet.

En la Tabla 2-3 se muestra la evaluación por el método de porcentaje ponderado para la selección de la red ideal para el acceso a servicios a realidad virtual.

Tabla 2-3: Selección de la red para acceso a servicios de realidad virtual a través de evaluación ponderada

Criterio (peso %)	**Red de cloud computing**		**Red centrada en el servidor**		**Red de Edge computing**		**Red privada virtual**	
	Puntuaje	Ponderado	Puntuaje	Ponderado	Puntuaje	Ponderado	Puntuaje	Ponderado
Flexibilidad de acceso (15%)	5	0.75	4	0.6	3	0.45	4	0.6
Bajo coste (20%)	4	0.8	2	0.4	1	0.2	2	0.4
Latencia baja (20%)	3	0.6	4	0.8	5	1	2	0.4
Acceso al servicio a través de internet (20%)	5	1	4	0.8	3	0.6	4	0.8
Factibilidad de implementación dentro de la red de la ESPOCH (25%)	2	0.5	5	1.25	2	0.5	4	1
Total		3.65		3.85		2.75		3.2

Realizado por: Huilcamaygua, Cinthya, 2023

La elección de la red centrada en el servidor por este método de porcentaje ponderado arroja que es la opción ideal para aplicarlo en el contexto de una red para acceso a servicios de realidad virtual y sobre todo es compatible con la infraestructura existente de la institución.

3.5.3.1 Escenario de red básico para la definición de elementos

Teniendo muy claro el tipo de red a utilizar se debe tener en cuenta elementos infaltables para el diseño del sistema como los mencionados anteriormente. Se ha definido un escenario base como punto de partida y que se modifica dependiendo de las necesidades de conectividad para cada protocolo, tal como se observa en la Ilustración 2-3.

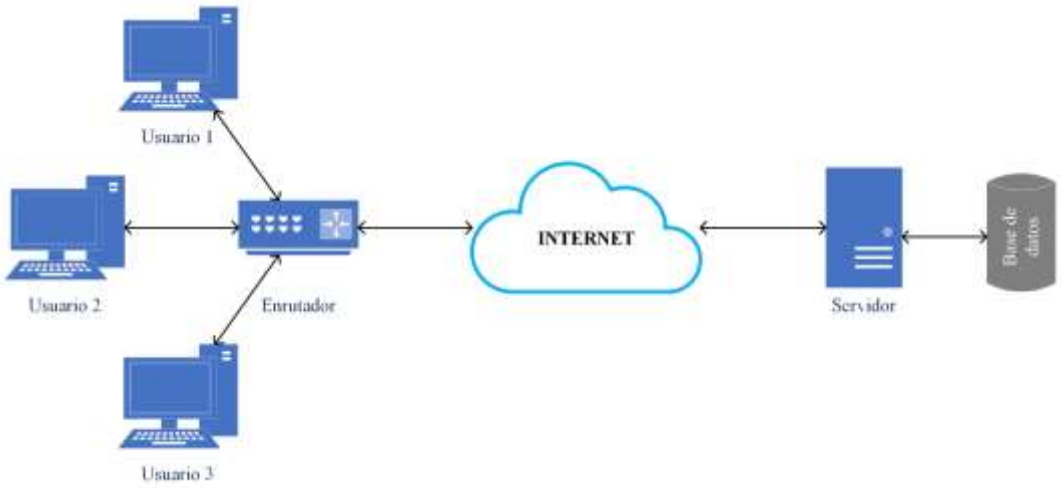


Ilustración 2-3: Escenario base para el acceso a servicios de realidad virtual.

Este sistema no cuenta con medidas de seguridad como encriptación de los datos o autenticación de usuarios porque el objetivo es enfocarse netamente en el rendimiento de los dos protocolos sin el peso adicional de las medidas de seguridad que podrían afectar los tiempos de respuesta. Además de esto, se puede evaluar la capacidad de ambos protocolos para manejar diferentes cargas de tráfico y la escalabilidad de la red. Esto permite identificar posibles limitaciones o cuellos de botella en términos de latencia y rendimiento cuando se trata de un gran número de conexiones o transmisiones simultáneas.

3.5.4 Selección del servidor centralizado

Lo más adecuado para este proyecto sería utilizar la infraestructura de la institución. Sin embargo, dado que el sistema está diseñado para una implementación futura, no se dispone de acceso a dicha infraestructura en la actualidad. Las opciones más viables para configurar el escenario de

red consisten en utilizar un servidor alojado en la nube. Esto se debe a que no es factible optar por un escenario en red local, dado que es necesario que los estudiantes puedan conectarse a Internet para enviar los datos al servidor.

La elección de un servidor alojado en la nube presenta ventajas significativas, ya que permite el acceso remoto desde cualquier ubicación con conexión a Internet. Esto resulta especialmente beneficioso para el propósito del proyecto, ya que los estudiantes pueden acceder al servidor y enviar sus datos de manera conveniente desde cualquier lugar y dispositivo con acceso a la web.

En contraste con lo anterior, un escenario de red local requeriría que los estudiantes estén físicamente presentes en la misma red local para enviar los datos al servidor, lo cual podría limitar la flexibilidad y la accesibilidad del sistema. Además, usar una máquina virtual en una red local puede ser útil para comprobar la compatibilidad de todos los elementos de la red en una primera instancia y luego implementarlo en el escenario definitivo.

La adopción de la nube ofrece una solución más práctica y escalable, asegurando la disponibilidad y conectividad necesarias para el intercambio de datos de manera eficiente. Dicho de otra manera, la comunicación en cualquier lugar, en cualquier momento desde cualquier dispositivo con la sola conexión a Internet son características importantes que están alineadas con las características de la infraestructura de la institución y también con un servidor alojado en la nube.

- Población

Existen diferentes opciones de servidores alojados en la nube, se han seleccionado todos los servidores que ofrecen servicios gratuitos en la nube tanto sus características más destacables como sus limitaciones, como se puede observar en la Tabla 3-3.

Tabla 3-3: Características de los servidores gratuitos alojados en la nube.

Servicio	Características Gratis Destacadas	Limitaciones
AWS Free Tier	12 meses de uso gratuito de algunos servicios	Límites de recursos específicos por servicio
	750 horas mensuales de cómputo de instancia EC2 t2.micro	Uso limitado de datos y almacenamiento
	1 dirección IP elástica gratuita mientras la instancia siga ejecutándose	La dirección IP elástica debe ser utilizada, de lo contrario se cobra un rubro adicional
	5 GB de almacenamiento en Amazon S3	Sólo para nuevos clientes o cuentas recién creadas
Google Cloud Free Tier	$300 en créditos durante 12 meses	Uso limitado de ciertos servicios como BigQuery y Cloud Pub/Sub
	1 f1micro instancia de VM por mes	Algunos servicios solo disponibles en regiones específicas

	5 GB de almacenamiento en Google Cloud Storage	
Microsoft Azure Free	$200 en créditos durante 30 días y ofrece IP pública con sus instancias	Uso limitado de ciertos servicios como Azure Cosmos DB
	12 meses de uso gratuito de algunos servicios	Límites de recursos específicos por servicio y se cobra una tarifa para IP estática o reservada
	250 GB de almacenamiento en Azure Blob Storage	
Oracle Cloud Free Tier	2 VMs Linux y 2 VMs Windows permanentes y asigna IP pública a instancias por defecto	Uso limitado de ciertos servicios como Autonomous Database
	2 bases de datos Autonomous	Límites de recursos específicos por servicio y las direcciones IP estáticas tienen un costo asociado
	10 GB de almacenamiento en Oracle Cloud Storage	
Heroku Free	5501000 horas mensuales de Dynos	Uso limitado de ciertos addons
	Uso compartido de recursos en servidores	Limitaciones en la cantidad de aplicaciones
Vercel	Implementación y alojamiento de proyectos públicos	Uso compartido de recursos en servidores
	Rápida configuración y despliegue de aplicaciones	Sin soporte para proyectos privados

Realizado por: Huilcamaygua, Cinthya, 2023

- Selección de la muestra no probabilística

Una característica en particular para elegir el servidor en la nube es que pueda ofrecer una IP elástica o pública al cual dirigir el tráfico generado por las aplicaciones de realidad virtual. Algunas de las opciones lo ofrecen, pero tiene un costo o rubro adicional, por lo que lo ideal sería optar por uno que ofrezca de forma gratuita. De manera que todas estas opciones se valoran por el método de evaluación ponderada para determinar el ideal para este trabajo (Ver Tabla 4-3)

Tabla 4-3: Características de los servidores gratuitos alojados en la nube.

Criterio (peso %)	**AWS**		**Google Cloud**		**MS Azure Free**		**Oracle Cloud**		**Heroku free**		**Vercel**	
	Punt.	Pond.	Punt.	Pond.	Punt.	Pond.	Punt.	Pond.	Punt.	Pond.	Punt.	Pond.
Desempeño (20%)	5	1	4	0.8	3	0.6	3	0.6	3	0.6	4	0.8
Disponibilidad Global (10%)	5	0.5	4	0.4	4	0.4	3	0.3	3	0.3	4	0.4
Facilidad de uso (20%)	4	0.8	4	0.8	4	0.8	3	0.6	4	0.8	4	0.8
IP elástica o pública (50%)	5	2.5	3	1.5	3	1.5	3	1.5	2	1	1	0.5
Total		4.8		3.5		3.3		3		2.7		2.5

Realizado por: Huilcamaygua, Cinthya, 2023

La opción que resalta es la de AWS según la puntación exhibida de la evaluación realizada. El aspecto en el cual se destacó fue en el criterio de IP elástica, pues es el que tenía la capacidad de

asignar una dirección IP publica a la instancia y de forma gratuita. En la Tabla 5-3 se muestran las características de este servidor y es el que se ha ocupado para el levantamiento del escenario de red.

Tabla 5-3: Características de la instancia del servidor virtual AWS

Tipo de virtualización	Virtualización asistida por hardware
Tipo de instancia	t2.micro
Sistema Operativo	Ubuntu Server
Versión del SO	20.04 LTS
Virtual CPUs	1
RAM	1 GB
Almacenamiento	30 GB de disco SSD
Tiempo de ejecución de la instancia	750 horas / mes
Dirección IP elástica	18.219.82.107
Nombre del par de claves	keyprueba.pem

Realizado por: Huilcamaygua, Cinthya, 2023.

3.5.5 Selección del motor de RV

Dentro de la población para la selección del motor de realidad virtual existen 4 opciones que se detallan en la Tabla 6-3.

Tabla 6-3: Características de los motores de realidad virtual.

Característica	Unity 3D	Unreal Engine	SteamVR	Godot Engine
Tipo de Motor	Motor de desarrollo de juegos y apps	Motor de desarrollo de juegos y apps	Plataforma y SDK de realidad virtual	Motor de desarrollo de juegos y apps
Facilidad de Uso	Amigable y accesible	Potente pero curva de aprendizaje	Varía según la herramienta	Intuitivo y fácil de aprender
Gráficos	Buena calidad, se adapta al hardware	Excelente calidad y realismo	Depende del contenido y hardware	Buena calidad y personalizable
Comunidad	Gran comunidad y abundante documentación	Amplia comunidad y recursos	Comunidad en crecimiento	Comunidad en crecimiento
Plataformas Soportadas	Múltiples plataformas (PC, consolas, etc.)	Múltiples plataformas (PC, consolas, etc.)	PC, Steam VR	Múltiples plataformas (PC, consolas, etc.)
Licencia	Propietaria (3 opciones)	Propietaria	Propietaria	MIT License
Ecosistema	Amplio ecosistema de assets y plugins	Amplio ecosistema de assets y plugins	Integrado con Steam y SteamVR	Creciente ecosistema de addons

Realizado por: Huilcamaygua, Cinthya, 2023.

- Selección de la muestra no probabilística

En la tabla 7-3 se muestran los puntajes obtenidos de la evaluación de la población de motores de realidad virtual de acuerdo con las características más importantes para el trabajo.

Tabla 7-3: Evaluación ponderada entre los diferentes motores de realidad virtual.

Criterio (Peso %)	Unity 3D		Unreal Engine		SteamVR		Godot Engine	
	Punt.	Pond.	Punt.	Pond.	Punt.	Pond.	Punt.	Pond.
Facilidad de Uso (20%)	5	1	3	0.6	4	0.8	4	0.8
Gráficos (15%)	4	0.6	5	0.75	3	0.45	4	0.6
Comunidad activa (15%)	5	0.75	4	0.6	3	0.45	3	0.45
Plataformas Soportadas (15%)	5	0.75	5	0.75	4	0.6	5	0.75
Código Abierto (5%)	1	0.05	1	0.05	1	0.05	5	0.25
Disponibilidad de versión gratuita (15%)	4	0.6	4	0.6	4	0.6	5	0.75
Ecosistema (15%)	5	0.75	5	0.75	4	0.6	4	0.6
Total ponderado	29	4.5	27	4.1	23	3.55	30	4.2

Realizado por: Huilcamaygua, Cinthya, 2023.

El motor de realidad virtual más apropiado para el proyecto fue Unity 3D según los resultados del método de evaluación ponderada, a su facilidad de uso, soporte de múltiples plataformas, gran comunidad de desarrolladores y la disponibilidad de una versión gratuita que ofrece un paquete completo para muchos proyectos de realidad virtual. Aunque el criterio de código abierto es importante para el trabajo, aun así, se lo ha seleccionado por los resultados obtenidos.

3.5.6 Selección de la de la base de datos

La selección de la base de datos es algo muy importante ya que es el lugar donde se van a alojar los datos de la aplicación de realidad virtual para ser usados posteriormente por la aplicación del profesor. Al no tener un escenario desarrollado con todos los elementos, aún se desconoce el tipo de datos a obtener y es posible obtener una gran cantidad de información y se planea que el proyecto escale en número de usuarios, para lo cual es ideal orientarse por una base de datos NoSQL. Se debe agregar que, las bases de datos NoSQL se pueden ejecutar en máquinas que poseen pocos recursos, lo que es ideal para aprovechar el rendimiento de la instancia AWS.

- Población

Las bases de datos NoSQL no son tan nuevas como parecen, pero han logrado sobresalir en los últimos años por su facilidad en el almacenamiento de datos, entre otros aspectos. La población

de las bases de datos no relacionales son MongoDB y Cassandra. En la Tabla 8-3 se puede observar las distintas características que tienen.

Tabla 8-3: Características de las bases de datos NoSQL.

Característica	MongoDB	Cassandra
Modelo de Datos	Documentos (JSON/BSON)	Columnas (wide column store)
Escalabilidad	Escalabilidad horizontal fácil	Escalabilidad horizontal fácil
Lenguaje de Consultas	MongoDB Query Language (MQL)	CQL (Cassandra Query Language)
Flexibilidad de Esquema	Esquema flexible, sin estructura fija	Esquema flexible, sin estructura fija
Replicación y Tolerancia a fallos	Soporte para replicación y tolerancia a fallos	Soporte para replicación y tolerancia a fallos
Distribución de Datos	Distribución automática de datos	Distribución automática de datos
Transacciones	Soporte para transacciones a nivel de documento	Soporte para transacciones limitado
Búsquedas Complejas	Soporte para consultas complejas	Limitaciones en consultas complejas
Escritura de Datos	Gran rendimiento de escritura	Gran rendimiento de escritura
Comunidad y Soporte	Gran comunidad y abundante documentación	Comunidad en crecimiento
Casos de Uso	Aplicaciones web, móviles, IoT, analíticas	Aplicaciones distribuidas a gran escala, series temporales

Realizado por: Huilcamaygua, Cinthya, 2023.

- Selección de la muestra no probabilística

En la Tabla 9-3 se evalúan las características de ambas bases de datos MongoDB y Cassandra por el método de evaluación ponderada.

Tabla 9-3: Selección de la base de datos mediante el método de evaluación ponderada.

Criterio	Peso %	MongoDB		Cassandra DB	
		Puntaje	Ponderación	Puntaje	Ponderación
Modelo de Datos	15	5	0.75	4	0.6
Escalabilidad	10	5	0.5	4	0.4
Flexibilidad de Esquema	10	4	0.4	3	0.3
Replicación y Tolerancia a fallos	10	5	0.5	3	0.3
Búsquedas Complejas	10	5	0.5	3	0.3
Escritura de Datos	15	5	0.75	3	0.45
Casos de uso	15	5	0.75	3	0.45
Comunidad y Soporte	15	5	0.75	4	0.6
Total			4.9		3.4

Realizado por: Huilcamaygua, Cinthya, 2023.

En esta evaluación ponderada, MongoDB resultó ser el mejor protocolo por distintas razones, pues, una de las más importantes es la del modelo de datos porque permite manejar datos con

esquemas flexibles, lo que simplifica la adaptación a cambios en la estructura de datos sin afectar la integridad.

3.5.7 Selección del protocolo de comunicación IoT

En el caso de la elección del protocolo de comunicación, se tiene que tomar en cuenta todos los elementos anteriores, como la red centrada en el servidor, la utilización del motor de realidad virtual, y la base de datos. Todos estos aspectos tienen que ser compatibles con el protocolo.

3.5.7.1 Análisis y comparación de protocolos IoT

En la Tabla 10-3 se muestra las características de cada protocolo de comunicación IoT. Estos protocolos fueron comparados para identificar la opción más idónea para su implementación en este proyecto. Además, se incorporan el protocolo HTTP y WebSocket en este análisis, ya que, a pesar de no ser un protocolo específico para IoT, es compatible con numerosas funciones de la Internet de las cosas.

Tabla 10-3: Características de los protocolos IoT

Protocolo	Transporte	Seguridad	Modelo de Comunicación	Uso de recursos	Latencia	Soporte QoS	Centralizado
MQTT	TCP/IP	TLS/SSL	Publicación - suscripción	Bajo	Baja	Sí	si
CoAP	UDP	DTLS	Solicitud-respuesta	Bajo	Baja	No	no
HTTP/HTTPS	TCP/IP	SSL/TLS	Solicitud-respuesta	Alto	Alta	No	si
AMQP	TCP/IP	TLS/SSL	Publicación - suscripción, punto a punto, multicast	Medio a Alto	Baja	Sí	no
WebSocket	TCP/IP	SSL/TLS	Bidireccional	Medio	Baja	No	si
XMPP	TCP/IP	TLS/SSL	Solicitud-respuesta	Medio	Baja	no	si

Realizado por: Huilcamaygua, Cinthya, 2023.

3.5.7.2 Selección de la muestra no probabilística del protocolo IoT

En la Tabla 11-3 se encuentran los puntajes obtenidos de cada protocolo según el criterio asignado en la tabla.

Tabla 11-3: Selección del protocolo IoT bajo el método de evaluación ponderada.

Criterio	Peso %	MQTT		CoAP		HTTP		AMQP		WebSocket		XMPP	
		Punt.	Pond.	Punt.	Pond.	Punt.	Pond.	Punt.	Pond.	Punt.	Pond.	Punt.	Pond.
Transporte	10	5	0.5	4	0.4	5	0.5	5	0.5	5	0.5	4	0.4
Compatibilidad Unity	20	4	0.8	3	0.6	5	1	3	0.6	5	1	3	0.6

Seguridad	20	5	1	3	0.6	5	1	5	1	4	0.8	5	1
Uso de recursos	10	5	0.5	4	0.4	1	0.1	3	0.3	4	0.4	3	0.3
Compatibilidad MongoDB	20	5	1	2	0.4	5	1	3	0.6	5	1	3	0.6
Facilidad de Uso	10	5	0.5	3	0.3	5	0.5	3	0.3	5	0.5	4	0.4
Soporte AWS	10	5	0.5	4	0.4	5	0.5	5	0.5	5	0.5	5	0.5
Total			4.8		3.1		4.6		3.8		4.7		3.8

Realizado por: Huilcamaygua, Cinthya, 2023.

El protocolo MQTT utilizado en el ámbito de la Industria, resulta ser el más acertado para el proyecto, y utiliza el modelo de publicación / suscripción en donde los datos son recibidos por un bróker bajo un tópico, y si se solicita datos bajo el mismo tópico, el bróker otorga el acceso a esa información.

La comunicación se vuelve muy rápida porque este intermediario no se encarga de redirigir o de enviar el tráfico hacia su destino, sino que la información llega y éste le concede acceso a quien se suscriba al mismo tópico. Además, se conoce que no encripta por defecto, pero una de las medidas de seguridad es agregar TLS. El tamaño del paquete es relativamente pequeño a comparación de su contraparte HTTP, lo que agiliza el envío de mensajes y es apto para entornos de ancho de banda estrecho. Al ser TCP el protocolo de transporte, se garantiza el envío de los mensajes hasta el bróker lo que resulta excelente para ambientes donde existen interferencias, por lo tanto, puede ser tolerante a fallos que ocurran en el medio de comunicación.(Al-Fuqaha et al., 2015)

Además, en esta evaluación se puede observar que otros protocolos de comunicación como HTTP y Websocket también son útiles para ser implementados en el presente trabajo de investigación dado que tuvieron una puntuación de 4.6 y 4.7 respectivamente.

3.5.8 Selección del intermediario de la comunicación o middleware

De acuerdo con el punto anterior, se llegó a la conclusión que el mejor protocolo de comunicación es MQTT que utiliza un modelo de comunicación centralizado usando un bróker como intermediario en la comunicación. La función del bróker es la de intercambiar información por medio de la publicación y suscripción de los datos.

- Población

Existe una gran variedad de intermediarios bróker para el protocolo MQTT, los más destacables se encuentran en la Tabla 12-3 en el cual se demuestran las características que tienen cada uno de acuerdo con varios criterios.

Tabla 12-3: Middlewares para la comunicación del protocolo MQTT y sus características.

Broker	Mosquitto	HiveMQ	EMQ X	IBM MessageSight	VerneMQ	CloudMQTT
Tipo	Código abierto	Comercial	Código abierto	Comercial	Código abierto	Servicio en la nube
Versión Gratuita	Sí	Gratuito y de paga	Gratuito y con licencia	No	Sí	Sí (Planes Gratis)
Escalable	Moderada	Alta	Alta	Alta	Alta	Variable
Protocolos Admitidos	MQTT v3.1, MQTT v3.1.1, MQTT v5	MQTT v3.1, MQTT v3.1.1, MQTT v5	MQTT v3.1, MQTT v3.1.1, MQTT v5, CoAP, LwM2M, WebSocket, STOMP	MQTT v3.1, MQTT v3.1.1	MQTT v3.1, MQTT v3.1.1, MQTT v5	MQTT v3.1, MQTT v3.1.1
Medidas de Seguridad	Autenticación de usuarios, encriptación TLS/SSL	Autenticación de usuarios, encriptación TLS/SSL	Autenticación de usuarios, encriptación TLS/SSL	Autenticación de usuarios, encriptación TLS/SSL	Autenticación de usuarios, encriptación TLS/SSL	Autenticación de usuarios, encriptación TLS/SSL
Ámbito	General	Empresarial	Doméstico, empresarial, computación en la nube	Empresarial	General	General
Lenguaje	C	Java	Erlang	-	Erlang	Ruby, Phyton, Node.js, Java, PHP
Conexiones Máximas	1000	25 (versión gratuita) y hasta 10 millones en la de paga	10000 versión gratuita y hasta 1 millón en la versión Enterprise	Variable	Variable	10000
Otras características	Ligero, fácil de usar, soporte para websockets	Clustering, alta disponibilidad, extensibilidad	Soporte de clústeres, alta disponibilidad	Alta disponibilidad, soporte empresarial	Clústeres, alta disponibilidad, fácil de extender	Fácil configuración

Realizado por: Huilcamaygua, Cinthya, 2023.

- Selección no probabilística de la muestra

Lo ideal sería que el bróker elegido sea compatible con MongoDB para guardar la información. De acuerdo con la tabla de las características de los intermediarios propuestos, se ha creado la Tabla 13-3 y se han añadido 2 criterios adicionales que son importantes para el presente trabajo.

Tabla 13-3: Selección del middleware MQTT bajo el método de evaluación ponderada.

Criterio	Peso %	Mosquitto		HiveMQ		EMQ X		IBM MessageSight		VerneMQ		CloudMQTT	
		Punt.	Pond.	Punt.	Pond.	Punt.	Pond.	Punt.	Pond.	Punt.	Pond.	Punt.	Pond.
Código Abierto	10	5	0.5	1	0.1	5	0.5	1	0.1	5	0.5	1	0.1
Versión Gratuita	10	5	0.5	1	0.1	5	0.5	1	0.1	5	0.5	5	0.5

Escalabilidad	10	3	0.3	5	0.5	5	0.5	5	0.5	5	0.5	2	0.2
Protocolos Admitidos	10	5	0.5	5	0.5	5	0.5	3	0.3	4	0.4	3	0.3
Medidas de Seguridad	10	5	0.5	5	0.5	5	0.5	5	0.5	5	0.5	5	0.5
Lenguaje	10	4	0.4	4	0.4	5	0.5	1	0.1	5	0.5	1	0.1
Interfaz amigable	10	4	0.4	5	0.5	5	0.5	5	0.5	5	0.5	3	0.3
Compatibilidad con MongoDB	20	5	1	5	1	4	0.8	1	0.2	4	0.8	5	1
Compatibilidad con AWS Free Tier	10	5	0.5	1	0.1	5	0.5	1	0.1	5	0.5	5	0.5
Total			4.6		3.7		4.8		2.4		4.7		3.5

Realizado por: Huilcamaygua, Cinthya, 2023.

EMQ X es la opción más adecuada para el presente trabajo de acuerdo con la evaluación realidad. Este bróker de comunicación permite la entrega de datos de forma fiable, rápida y segura; maneja una gran cantidad de datos en tiempo real en el menor tiempo posible y además abarca hasta 10000 usuarios conectados en la versión gratuita de código abierto.

Un gran punto a su favor es que es un software de código completamente abierto que se encuentra alojado en GitHub donde una gran comunidad ayuda a su desarrollo, pero además de ello cuenta con una versión Enterprise que tiene más funciones que la versión gratuita que es una excelente opción para las empresas que deseen ampliar el número de usuarios y tener más funciones como el de una base de datos de respaldo para guardar los mensajes, esto es con un rubro adicional por estas funciones.

La versión de EMQ X Broker Open Source a instalar sería la última versión hasta el momento, y que sea compatible con Ubuntu 20.04. En su página web se muestra tal como la Ilustración 3-3 con varias opciones de descarga según la versión de bróker y sistema operativo deseado.

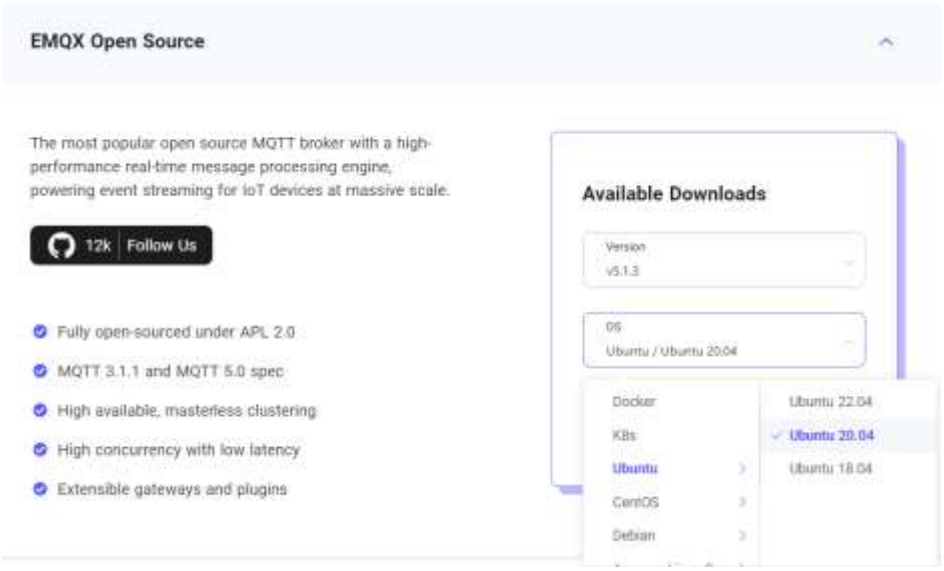


Ilustración 3-3: Versión de bróker utilizado para el trabajo de investigación.

Fuente: https://www.emqx.com/en/try?product=broker

3.6 Diseño de la red para acceso a servicios de realidad virtual

Una vez definidos los elementos para la red de acceso a servicios de realidad virtual, se diseñó el escenario para realizar la comunicación entre el usuario y el middleware de acuerdo con los requerimientos propuestos. En la Ilustración 4-3 se pueden observar todos los elementos interconectados elegidos, esto se tiene como un primer boceto, a medida que haya la necesidad de conectar otras aplicaciones o elementos adicionales, el diseño se modifica.

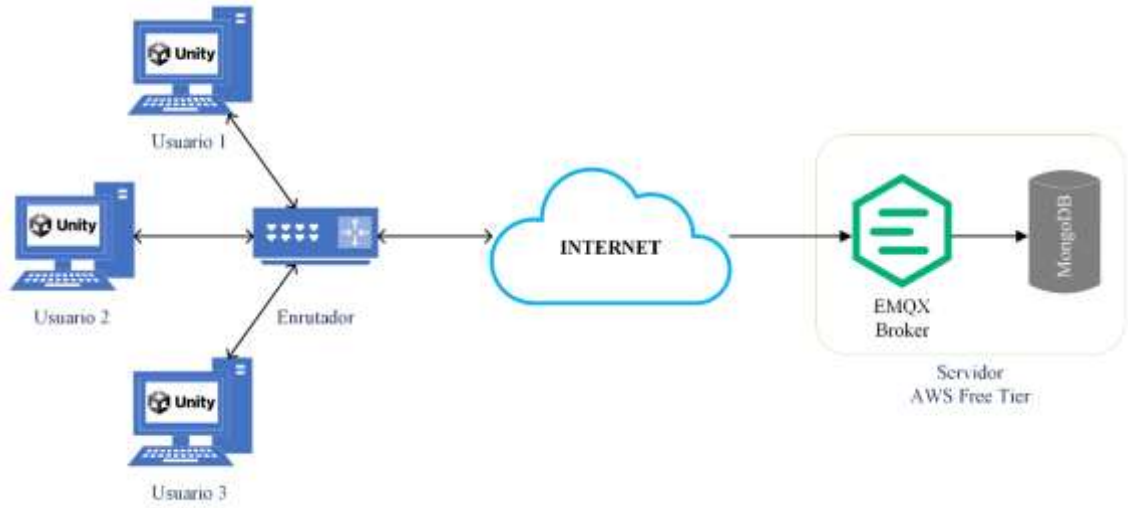


Ilustración 4-3: Primer boceto del diseño de la red para acceso a servicios de realidad virtual orientado al IoT con MQTT

Realizado por: Huilcamaygua, Cinthya, 2023.

En la Ilustración 4-3 se puede observar la conexión directa de EMQ X Broker con MongoDB. El bróker incluye plugins que permiten la conexión a bases de datos y otros sistemas externos para ampliar sus funcionalidades, como la conectividad nativa con MongoDB para mensajes persistentes.

A pesar de la importancia de la conectividad nativa en EMQ X a MongoDB para el desarrollo de este sistema, la función de mensajes persistentes no es de utilidad en el marco de este trabajo de investigación porque se debe acceder a datos específicos que hayan sido guardados previamente, y esta función no lo ofrece (OASIS, 2019). Por eso se ha visto la necesidad de optar por una aplicación externa para gestionar o procesar los datos. Esto se puede lograr mediante una herramienta que interactúe con EMQ X Broker y MongoDB.

Lo más propicio para este trabajo fue crear algo similar a un cliente de EMQX, pero que éste pudiera gestionar todos los mensajes que le llegan y condicionarlo a lo que se desea en el proyecto. Esto se ha logrado mediante la integración de un lenguaje de programación como Node.js para crear una aplicación que se conecte a EMQX Broker como un cliente más, pero con la particularidad de gestionar o procesar la información y almacenarla en la base de datos. Asimismo, este cliente Node.js puede recuperar la información de la base de datos y actuar como un cliente MQTT que puede enviar mensajes.

De manera que el diseño final de esta red es la mostrada en la Ilustración 5-3 y está lista para ser implementada.

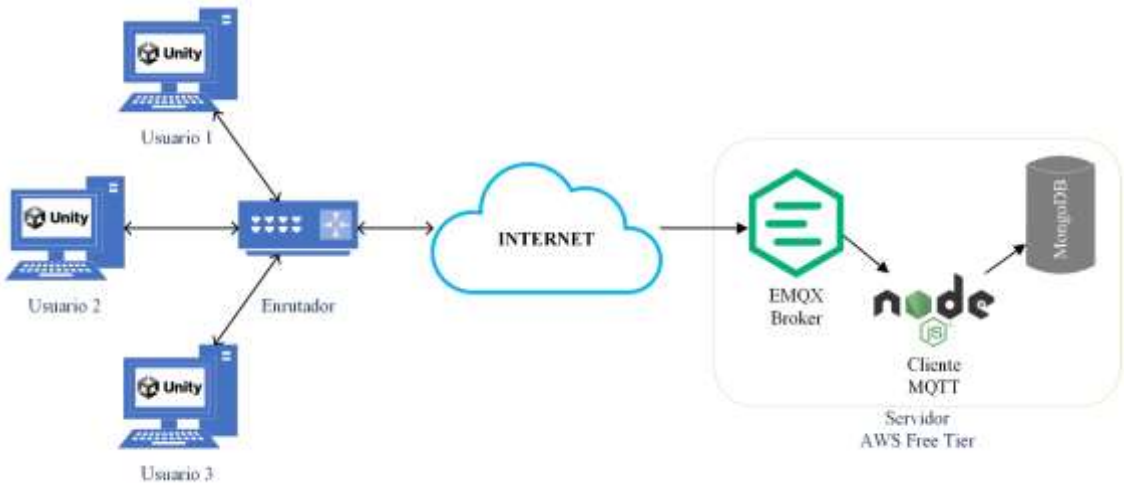


Ilustración 5-3: Diseño final de la red para acceso a servicios de realidad virtual orientado al IoT con MQTT

Realizado por: Huilcamaygua, Cinthya, 2023.

3.7 Implementación de la red para acceso a servicios de realidad virtual

3.7.1 Levantamiento del servidor

3.7.1.1 Creación de la cuenta AWS y configuración de la instancia

El primer paso para levantar el servidor virtual en AWS de forma gratuita se inicia con la creación de una cuenta en la página de AWS. La opción aparece en la página principal de AWS, tal como se puede ver en la Ilustración 6-3.

Ilustración 6-3: Página principal de AWS

Fuente: https://aws.amazon.com/es/what-is-aws/

Después de seleccionar la opción “Cree una cuenta de AWS”, se redirige a otra página para llenar un formulario de registro. Se deben introducir todos los datos que solicita el formulario para la creación de la cuenta, inclusive es necesario introducir los datos de una tarjeta de crédito para poder validar la cuenta. Este formulario se muestra en la Ilustración 7-3 a modo de una secuencia de 5 pasos empezando por el registro con correo y contraseña, luego está la información de contacto, la información de facturación, la confirmación de la identidad (una vez validada la tarjeta de crédito) y al final se selecciona el plan, que en este caso es el plan Basic gratis.

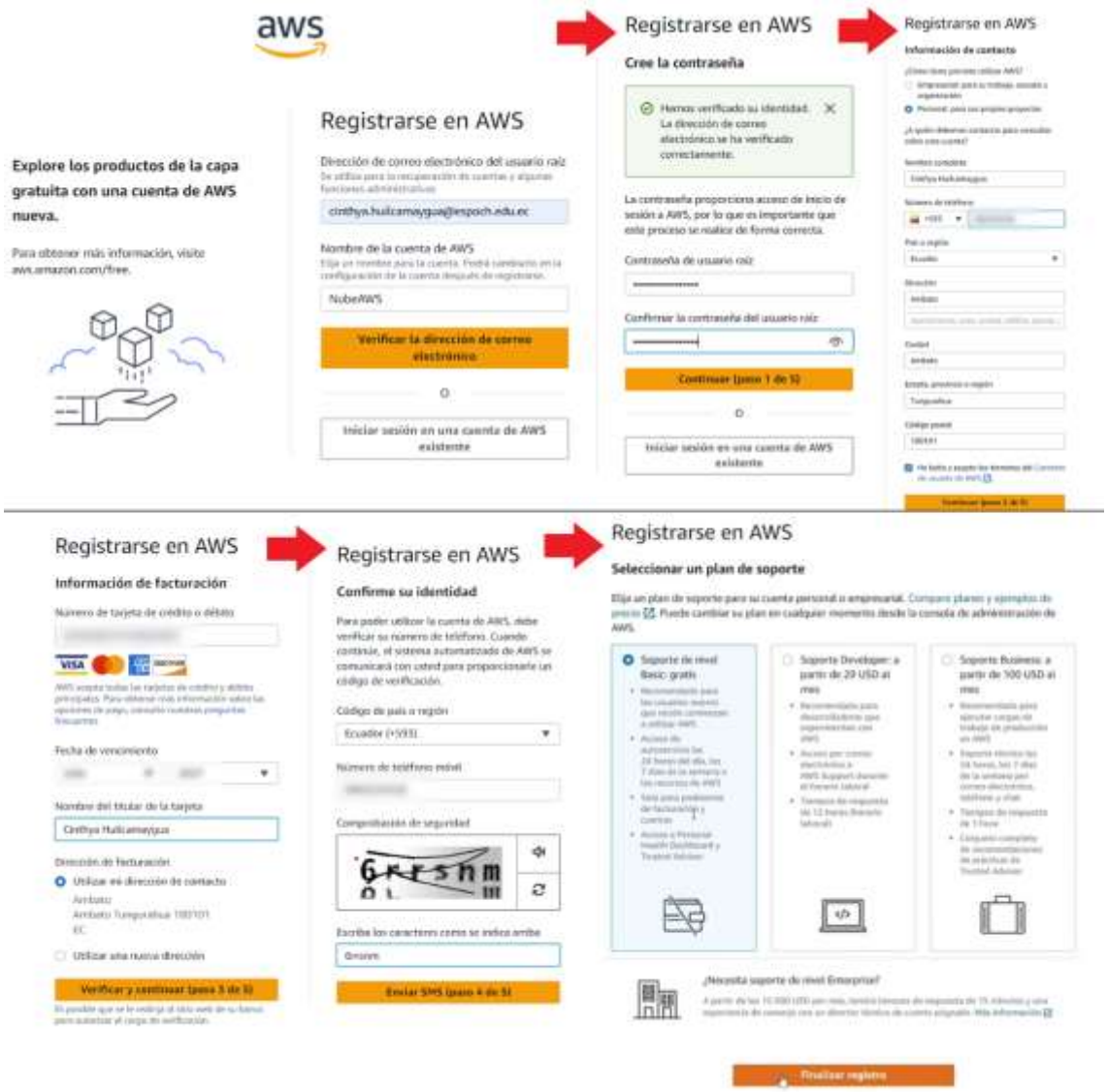


Ilustración 7-3: Registro en AWS

Fuente: https://portal.aws.amazon.com/billing/signup#/start/email

Realizado por: Huilcamaygua, Cinthya, 2023.

Una vez finalizado el registro, se muestra un botón llamado “Ir a la consola de administración de AWS” para iniciar la sesión y configurar la instancia, se debe acceder a la consola de AWS como usuario raíz y con el correo electrónico con el cual fue creada la cuenta. En la Ilustración 8-3 se muestra la ventana de inicio de sesión.

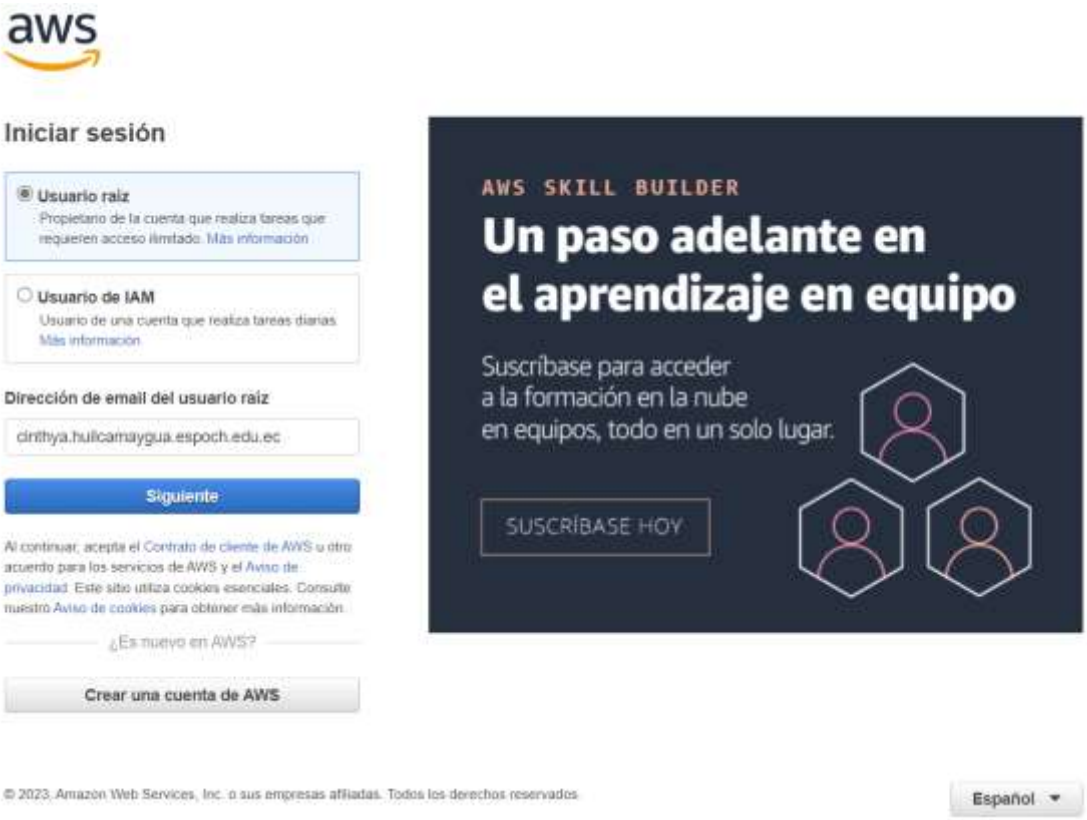


Ilustración 8-3: Interfaz de inicio de sesión en AWS.

3.7.1.2 Configuración de AWS

Una vez ingresado a la consola, en la misma página principal desplazándose hacia abajo se encuentra un menú llamado “Crear una solución”. Allí se encuentran varias funciones y la que se ejecutó fue “Lance una máquina virtual” que tiene una duración para su creación de 2 minutos, tal como se muestra en la Ilustración 9-3.

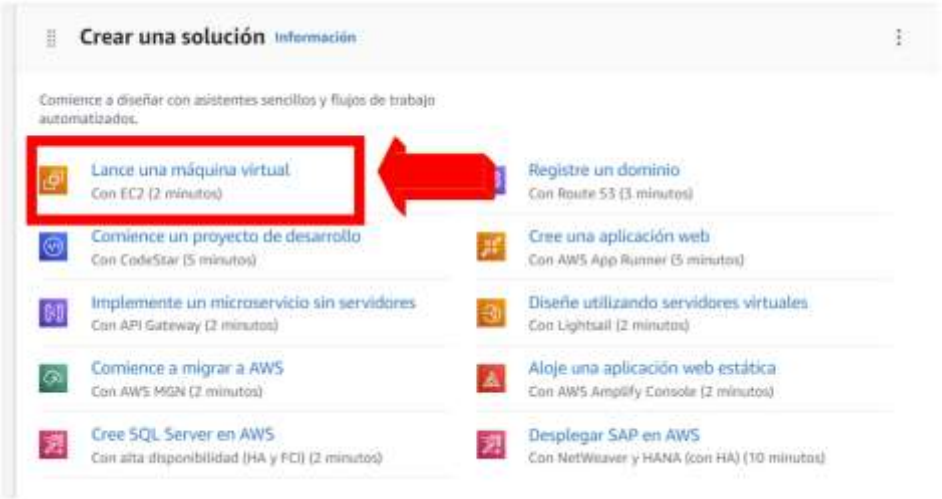


Ilustración 9-3: Interfaz de inicio de sesión en AWS.

Fuente: https://us-east-1.console.aws.amazon.com/ec2

Esta opción redirecciona enseguida a una página para la creación de la Instancia en donde se sigue una secuencia para lanzar la máquina virtual. Estos pasos son relativamente sencillos y consiste en:

- Colocar el nombre de la instancia (Ver Ilustración 10-3)
- Seleccionar el sistema operativo (Ver Ilustración 10-3)
- Tipo de instancia: t2.micro por defecto
- Creación de un par de claves (keyprueba.pem) para conectarse a través de SSH
- Configuración del grupo de seguridad con filtro de tráfico de protocolos
- Configuración de almacenamiento

- En detalles avanzados, habilitar la protección ante terminación accidental

- Revisión de final de la instancia

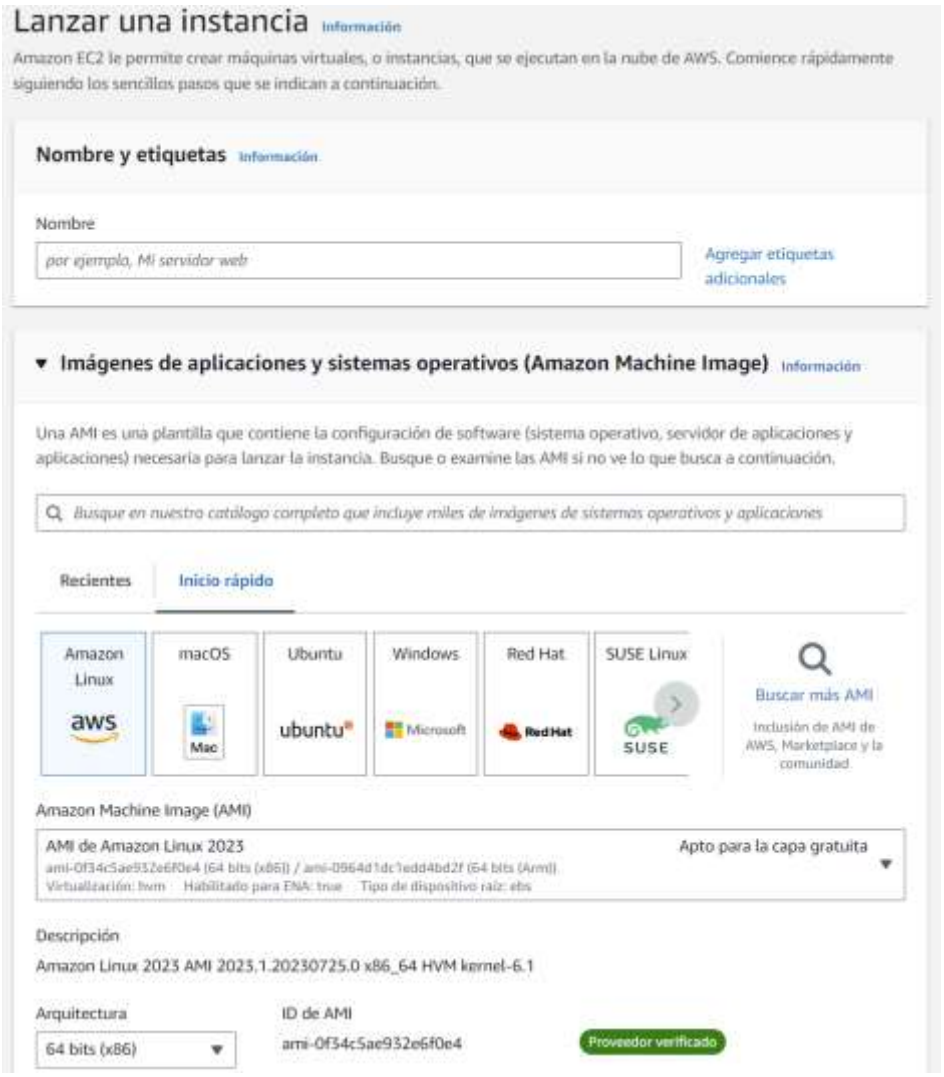


Ilustración 10-3: Interfaz de la opción Lanzar una instancia donde se muestran las primeras opciones de Nombre y selección de Sistema Operativo.

Fuente: https://us-east-1.console.aws.amazon.com/ec2

Después de configurar todas estas opciones, se tiene el resumen de la instancia donde se corrobora toda configuración y finalmente se lanza la instancia (Ver Ilustración 11-3)

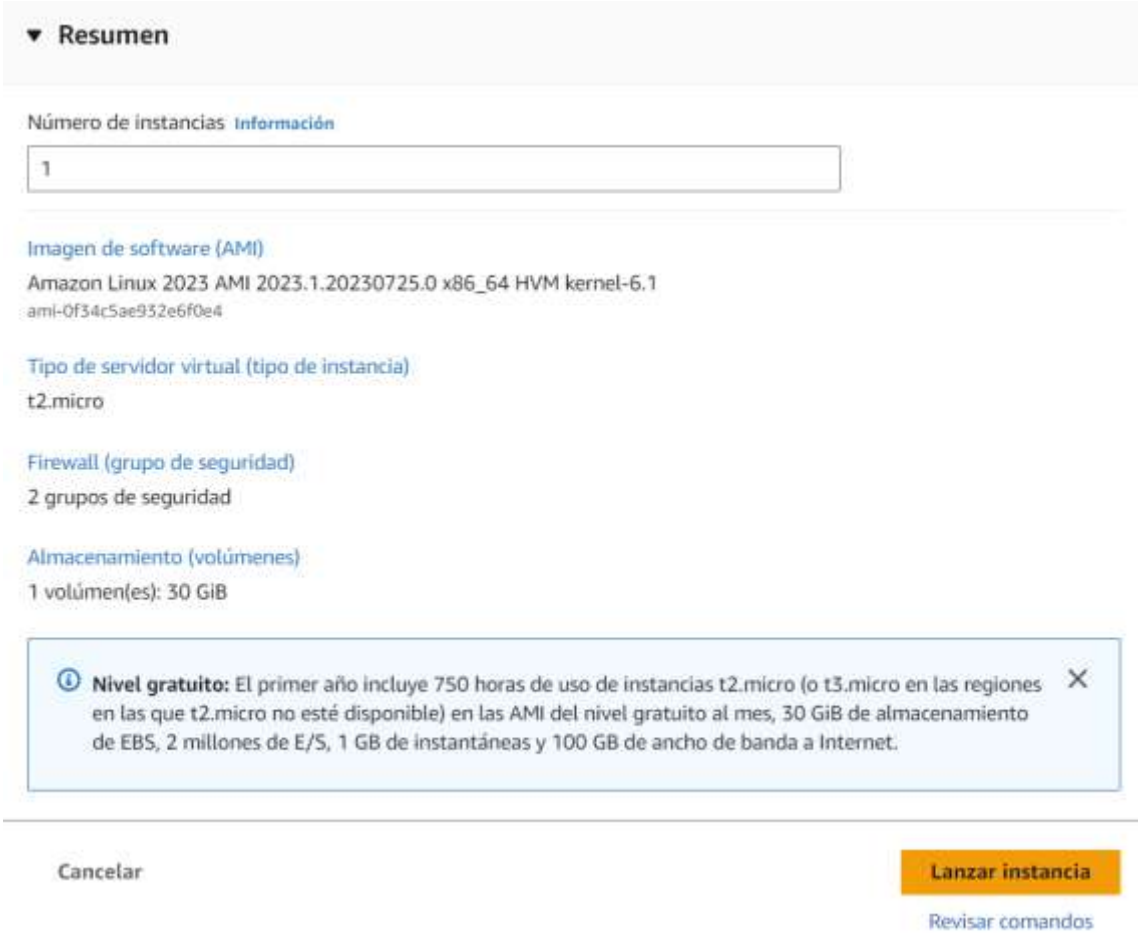


Ilustración 11-3: Resumen de la instancia

Fuente: https://us-east-1.console.aws.amazon.com/ec2

Es importante mencionar que, al lanzar la instancia también se descarga un par de llaves o claves que en este caso serviría para conectarse de manera remota por medio de SSH. La instancia se demora unos minutos hasta terminar de ser configurada hasta que su estado aparezca como En Ejecución, tal como se muestra en la Ilustración 12-3.

Ilustración 12-3: Resumen de la instancia

Fuente: https://us-east-1.console.aws.amazon.com/ec2

Para finalizar la configuración del servidor, se ha asignado una dirección IP elástica (fija) a la instancia creada para que la aplicación de RV apunte sus mensajes hacia el servidor, teniendo como resultado una dirección publica asignada automáticamente por AWS. Esto se realiza seleccionando entre las diferentes opciones del menú izquierdo dentro de la categoría Red y Seguridad está la opción Direcciones IP Elásticas. En esta ventana hay un botón llamado "Asignar la dirección IP elástica". Esto dirige a otra ventana en donde solamente se debe presionar el botón "Asignar" (Ver Ilustración 13-3) y se obtiene una dirección IP elástica.

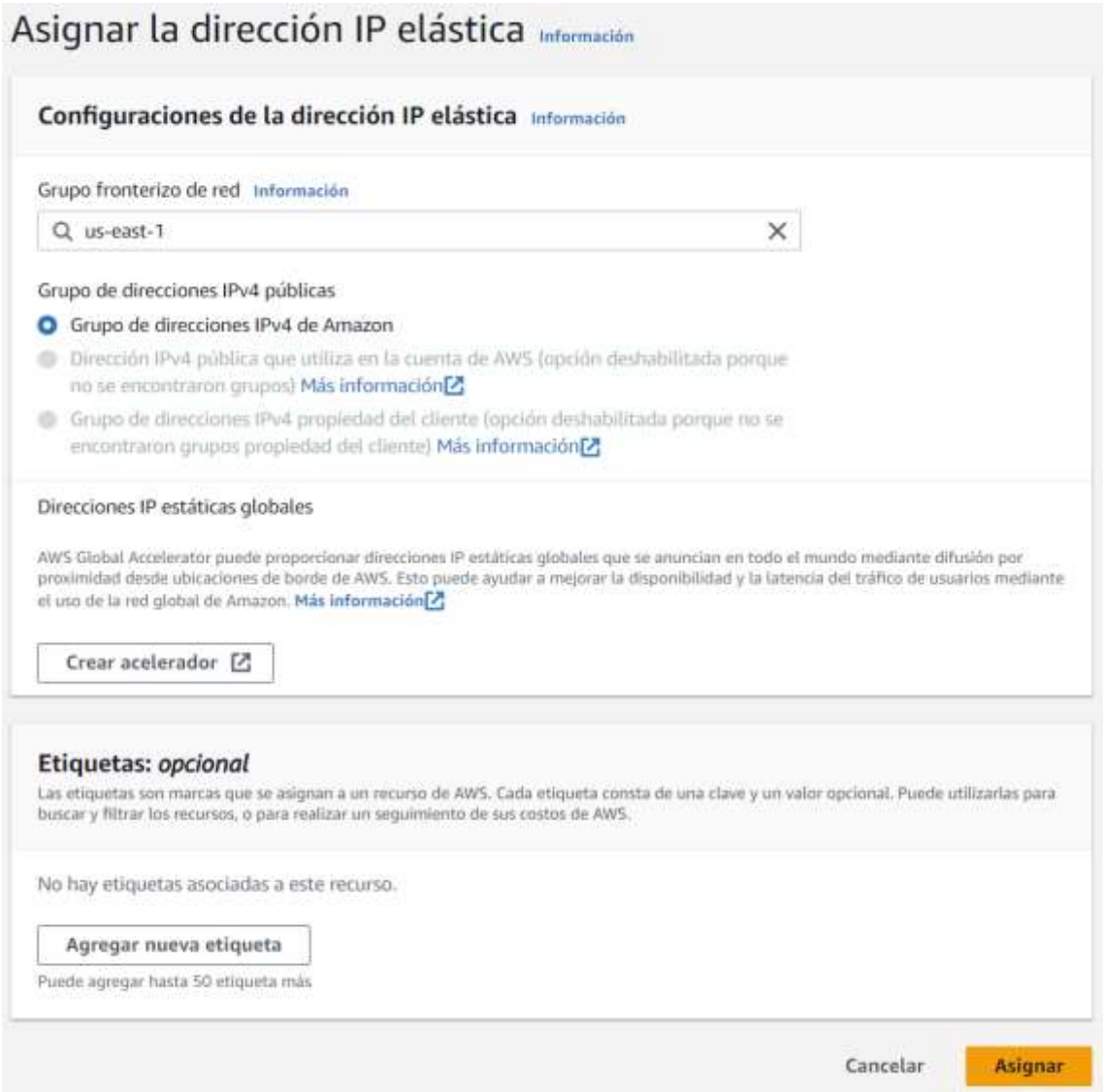


Ilustración 13-3: Creación de dirección IP elástica.

Fuente: https://us-east-1.console.aws.amazon.com/ec2

Luego de esto, aparece un mensaje en la ventana principal del Panel EC2 que dice “Asociar esta dirección IP elástica”. Este mensaje tiene un color verde que no pasa desapercibido como en la Ilustración 14-3.

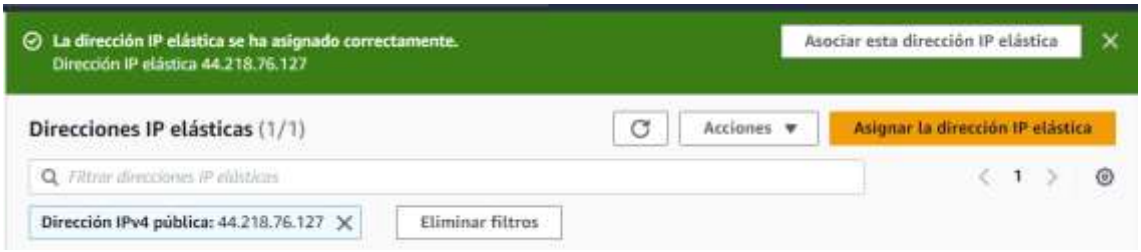


Ilustración 14-3: Dirección IP publica creada

Fuente: https://us-east-1.console.aws.amazon.com/ec2

En la asignación de la dirección IP elástica, se elige la única instancia creada con su respectiva dirección IP privada y fácilmente se asigna la instancia a esa IP publica, como se ve en la Ilustración 15-3.

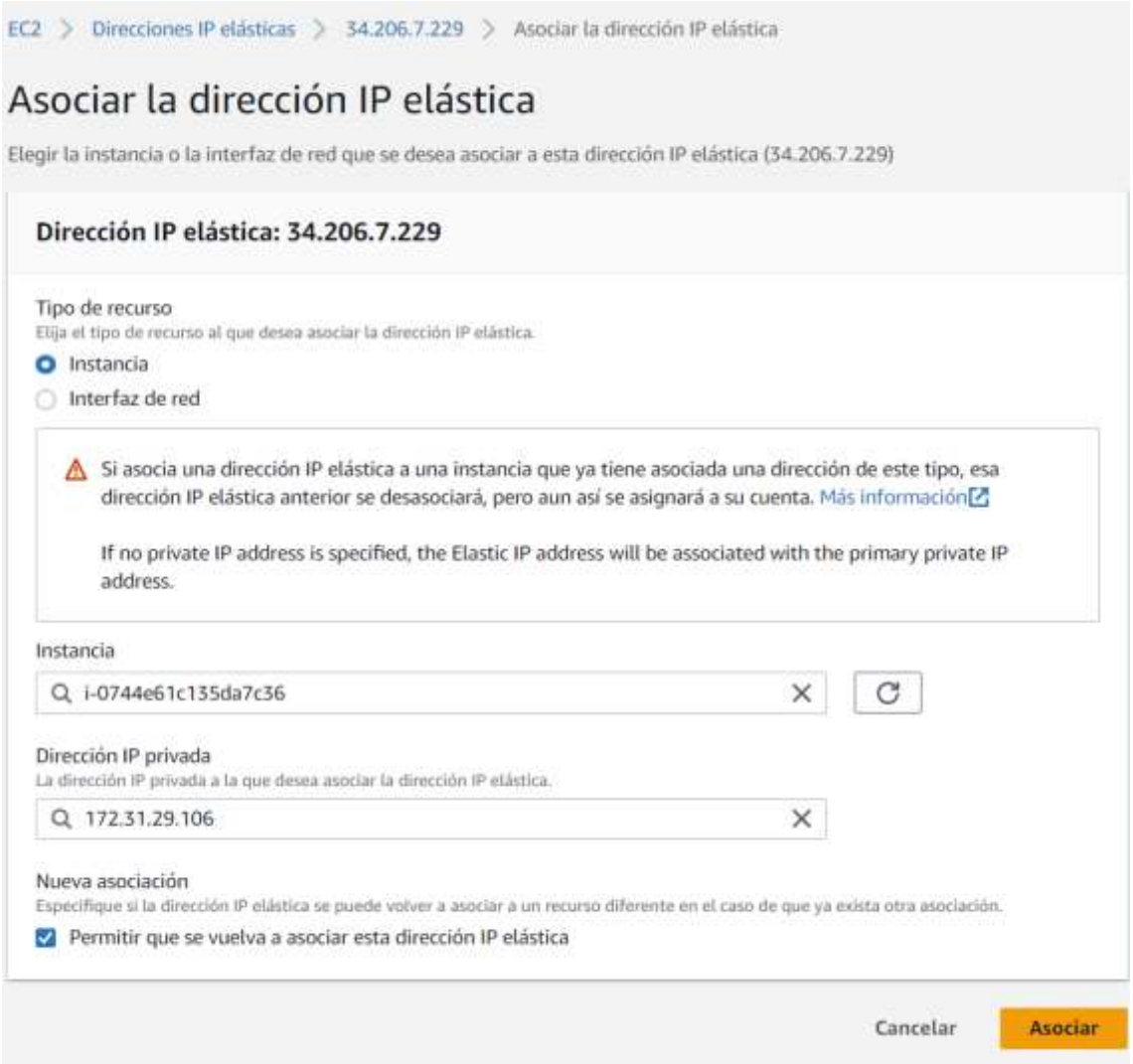


Ilustración 15-3: Asignación de la instancia a la dirección IP pública.

Fuente: https://us-east-1.console.aws.amazon.com/ec2

3.7.2 Instalación de EMQ X Bróker, MongoDB y Node.js en servidor

- Cliente SSH

Para la instalación de todos estos elementos, se lo puede realizar a través de una conexión SSH o también desde la misma consola que ofrece AWS.

Lo mejor es conectarse a través de SHH con el par de claves descargado en la creación de la instancia, esta información se la puede encontrar en la sección Instancias → Conectar. Allí aparece una ventana donde está detallada la información para conectar a través de un Cliente SSH, tal como se ve en la Ilustración 16-3.

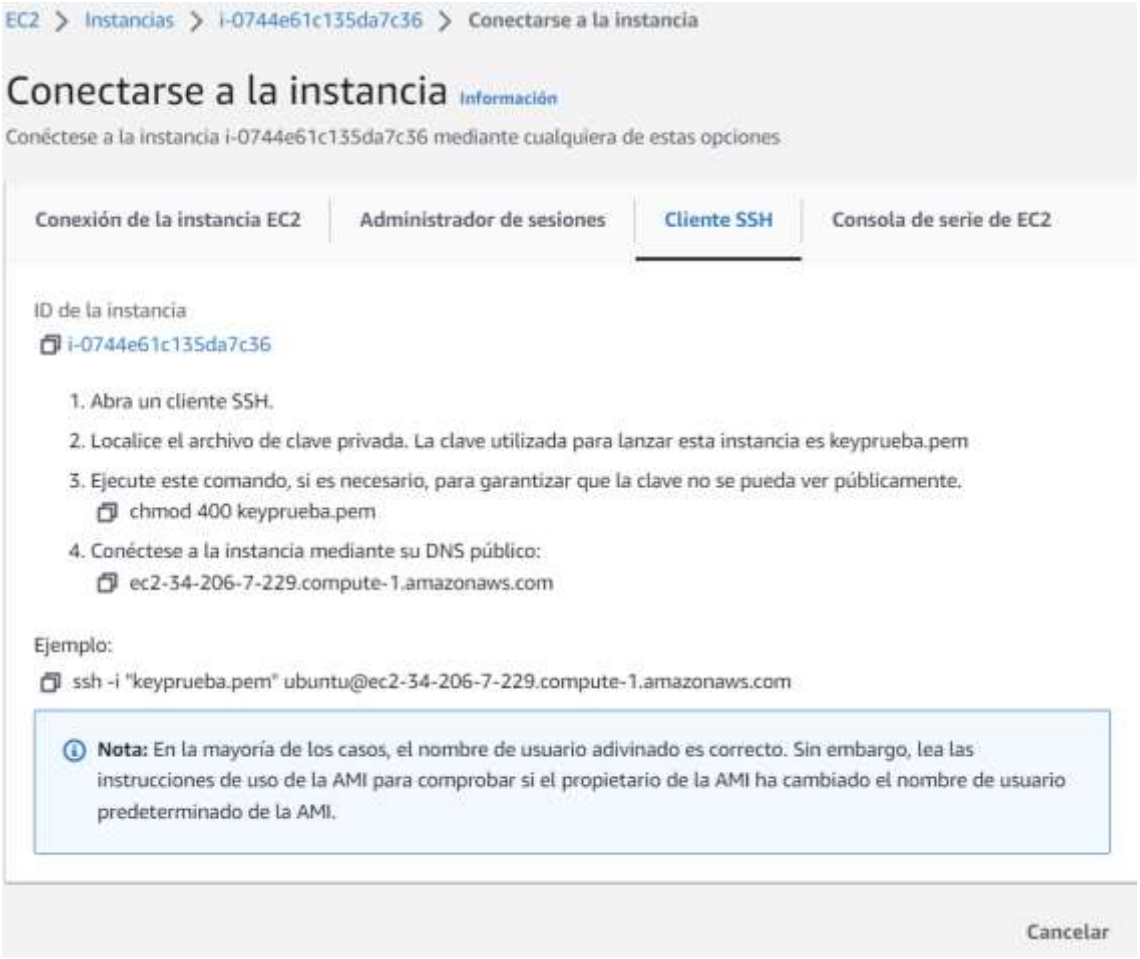


Ilustración 16-3: Pasos para la conexión al servidor a través de un cliente SSH

Fuente: https://us-east-1.console.aws.amazon.com/ec2

Cualquier software que permita la conexión SSH funciona para realizar cualquier cambio en la consola de la instancia AWS, inclusive se lo puede hacer desde AWS CloudShell.

3.7.2.1 Instalación de EMQ X Bróker

La instalación del se puede realizar netamente a través de la línea de comandos de Ubuntu con comandos que se encuentran en la página oficial de EMQX, todas las líneas de comandos estarán listas al elegir la versión de bróker y el sistema operativo, tal como en la Ilustración 17-3.

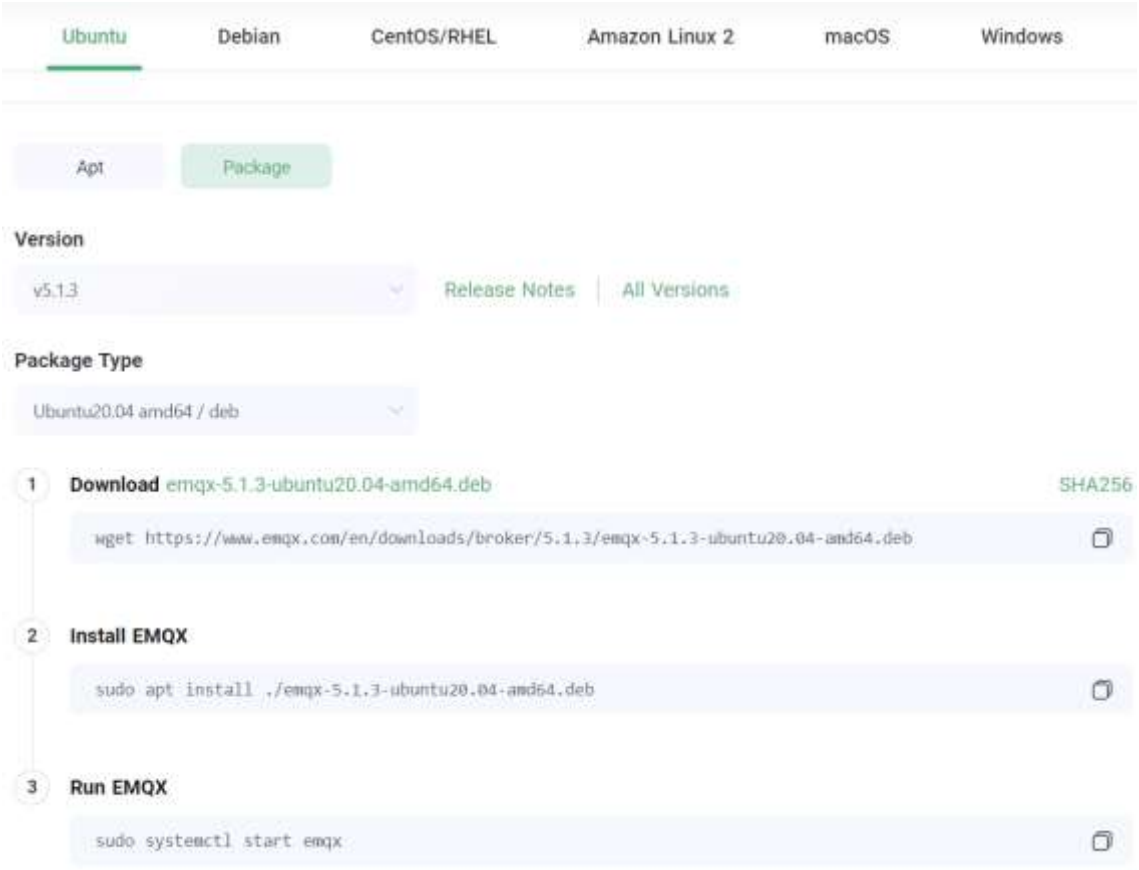


Ilustración 17-3: Descarga de EMQ X a través de líneas de comando.

Fuente: https://www.emqx.io/downloads?os=Ubuntu

- Instalación e inicio de MongoDB

Después de conectar el cliente SSH MobaXterm hacia la instancia de AWS, el primer paso fue instalar la base de datos MongoDB, escribiendo los comandos del Anexo A línea por línea.

Después de la instalación, se verificó si el sistema es systemd o init, con el siguiente comando, en este caso devuelve systemd.

```
ps --no-headers -o comm 1
```

Luego se escribe el comando:

```
sudo systemctl start mongod
```

Si ha fallado el inicio se tiene que recargar el Daemon, se ejecuta `sudo systemctl daemon-reload`

Después, escribimos la siguiente línea de comando para indicar que MongoDB está activo:

```
sudo systemctl status mongod
```

Si no se encuentra activo, se ejecuta el comando:

```
sudo systemctl enable mongod
```

3.7.2.2 Instalación de Node.js

La instalación del Cliente Node se lo realizó siguiendo los comandos mostrados a continuación:

```
git clone https://github.com/dennisdegreef/mqtt-mongo-recorder.git
cd mqtt-mongo-recorder
apt install npm
npm install
```

y para mantener activo el cliente node en segundo plano, se coloca el siguiente comando:

```
nohup node server.js &
```

La programación de la aplicación realizada en Node.js para gestionar la aplicación de prueba con MQTT se encuentra en el Anexo A.

La programación de la aplicación realizada en Node.js para gestionar la aplicación de prueba con HTTP se encuentra en el Anexo B.

3.8 Desarrollo de la aplicación de Realidad Virtual (RV)

El entorno de realidad virtual se ha diseñado en Unity 3D de tal manera que cumpla con los requisitos o requerimientos propuestos tanto por el tutor del presente trabajo de titulación como del investigador.

El componente principal del entorno 3D es el brazo robótico, con el que se interactúa y los movimientos realizados se pueden almacenar para visualizarlos posteriormente y también visualizarlos en tiempo real. Para esto, se han elaborado 2 aplicaciones: una para el profesor y la otra aplicación para dos estudiantes utilizando el mismo ambiente 3D, pero con diferentes funciones para cada usuario. En la aplicación del profesor, el usuario profesor puede ver los movimientos realizados en tiempo real por parte de los estudiantes, mientras que en la aplicación de los estudiantes van a existir 2 ambientes iguales con funciones para cada uno de los usuarios.

3.8.1 Definición de funciones e interacciones

La aplicación de realidad virtual tiene como objetivo la manipulación de variables en el ambiente 3D por parte del usuario, generando datos o información útil que será transmitida a través de internet y almacenada en una base de datos. Estos datos serían utilizados posteriormente por la aplicación del profesor para visualizarlos como movimientos en la aplicación de Unity 3D. Por ello, se desarrollaron 2 aplicaciones, una aplicación para 2 estudiantes y otra aplicación para un docente.

En la aplicación estudiantil la ventana principal tiene dos botones para que acceda el estudiante 1 o el 2. A continuación entraría a un laboratorio virtual en donde se encontrarían dos brazos robóticos (uno para cada estudiante) que se mueven según el ángulo que escoja el usuario. Con el movimiento de los ángulos del brazo robótico se puede modificar la posición por defecto hasta conseguir la deseada y guardar hasta 5 posiciones diferentes. Las 5 posiciones diferentes corresponden a 1 intento. Donde el total de intentos de la aplicación es de 3 intentos. Estos 3 intentos se envían con un botón hacia la base de datos a través del bróker y del cliente Node.js

Mientras que el profesor, en su entorno virtual es capaz de acceder a esos datos guardados en la base de datos y los puede visualizar tal y como el estudiante los guardó. La aplicación del profesor tiene la capacidad de ver los brazos de ambos estudiantes y así mismo, de ver los intentos que guardaron cada uno. Además, el profesor puede reiniciar el número de intentos para que los

estudiantes pueda realizar nuevamente la práctica de laboratorio y los valores guardados también se resetean.

3.8.2 Inicio del proyecto 3D

Primero, en el editor de Unity es esencial ajustar el entorno de juego y la resolución de la pantalla desde el inicio. Esto garantizará un trabajo sin contratiempos a lo largo del proceso de desarrollo de la aplicación. Dado que se busca que el entorno 3D sea ejecutado en una PC, este enfoque se justifica por la conveniencia que ofrece la PC para recopilar los datos necesarios.

Cabe recalcar que el presente trabajo fue pensado para ser utilizado como APK en un teléfono celular con la utilización de gafas de realidad virtual o Virtual VR Box para dar una sensación de inmersión, pero por motivos de análisis del sistema, la mejor opción es la de ejecutar el juego en un ordenador.

Para cambiar de plataforma en Unity se acudió a la sección File → Build Settings, donde aparece una ventana con varias opciones de plataformas, seleccionando la plataforma “PC, Mac & Linux Standalone”, dando click en “Bluid” como se observa en la Ilustración 18-3.

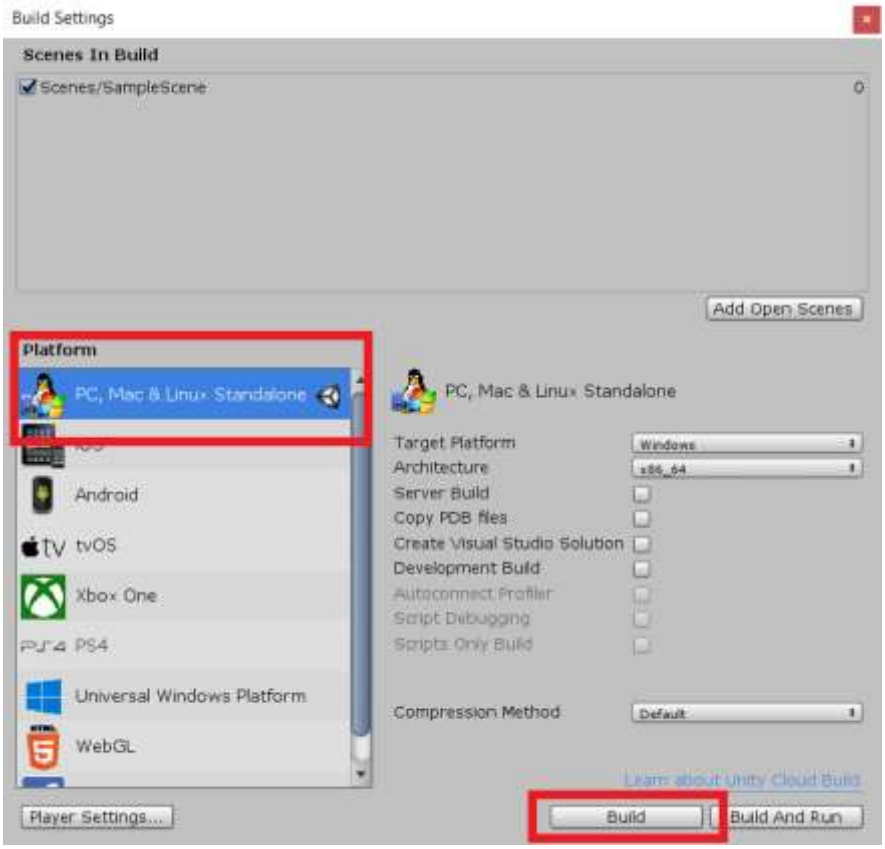


Ilustración 18-3: Selección de plataforma para el proyecto.

Realizado por: Huilcamaygua, Cinthya, 2023.

A pesar de que esta opción se encuentra seleccionada por defecto, se debe asegurar que la ventana en la que se está trabajando es la que se necesita. Luego se realizarían unas primeras interacciones con el servidor a partir de Unity. La primera interacción fue enviar un mensaje colocando indicaciones y un botón que envía el mensaje introducido en el recuadro de ingreso de texto, Esto se puede observar en la Ilustración 19-3.

Ilustración 19-3: Interfaz de pruebas para envió de mensajes hacia el bróker.

Realizado por: Huilcamaygua, Cinthya, 2023.

Con esto se comprueba que el primer boceto de la aplicación de RV es útil para enviar mensajes con el protocolo MQTT.

3.8.3 Ambiente 3D y brazo robótico

El entorno tridimensional se asemejaría a un laboratorio, en el cual se presentaría un componente con el que interactuar. El brazo robótico es un elemento que se puede encontrar en la mayoría de los laboratorios de las instituciones de educación superior, por lo que es idóneo para este contexto debido a su capacidad de ser manipulado a voluntad, según las necesidades del usuario. Otros dispositivos o elementos con los que se podría interactuar, como un tanque de agua o una válvula, también podrían ser considerados. Sin embargo, su integración en la aplicación de realidad virtual carece del atractivo o nivel de interacción como si lo posee el brazo robótico.

Se llevaron a cabo diversas pruebas piloto para la creación del entorno virtual en Unity. Esto se debió a que el desarrollo de este entorno se inició desde cero, y se tuvo que escoger entre muchos

Assets de brazos robóticos para poder moverlos de acuerdo con distintos ángulos. (Ver Ilustración 20-3)

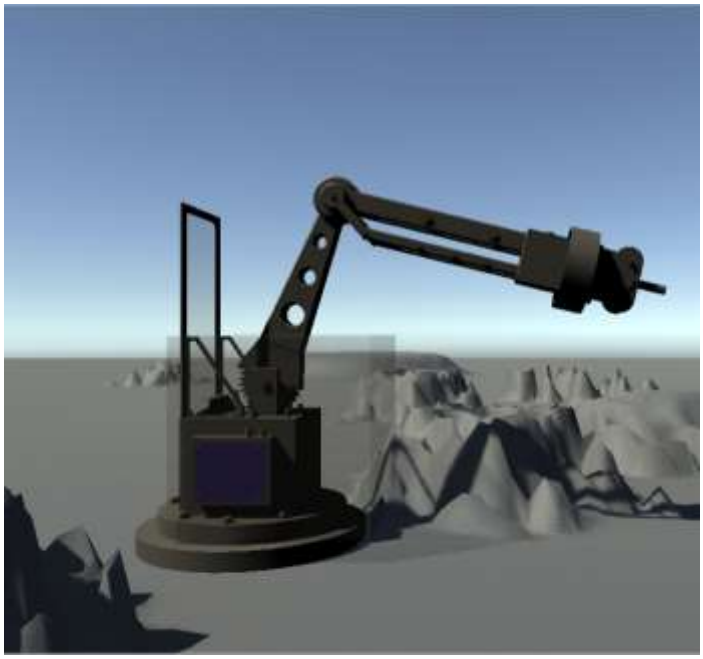


Ilustración 20-3: Asset de brazo robótico

Realizado por: Huilcamaygua, Cinthya, 2023.

Basándose en el Asset adquirido en la tienda de recursos de Unity, que representa un brazo robótico, se implementaron funciones para visualizar los movimientos en una ventana o menú flotante. Este panel interactivo permite la manipulación de los movimientos del brazo robótico. Esto se lo puede visualizar en la Ilustración 21-3.

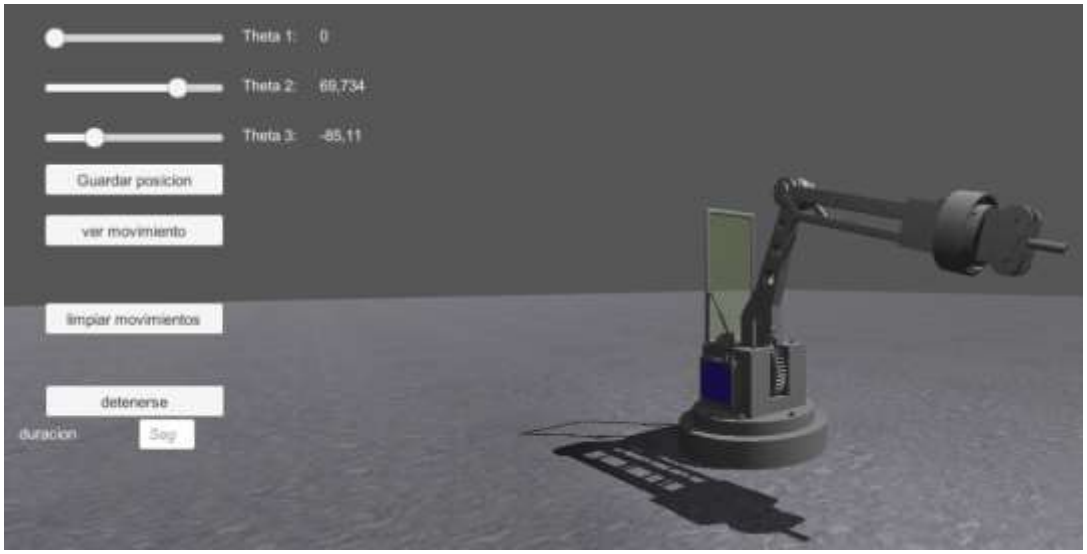


Ilustración 21-3: Panel interactivo del brazo robótico

Realizado por: Huilcamaygua, Cinthya, 2023.

3.8.4 Desarrollo de la aplicación de los estudiantes

Después de definir los movimientos del brazo, la interacción con el servidor y la transmisión de estos movimientos se llevarían a cabo mediante botones adicionales. Sin embargo, en términos de la representación visual del entorno, carece de un atractivo estético. La intención es crear la impresión de un laboratorio, lo cual implicaría la necesidad de ajustar el aspecto gráfico de la aplicación. Estos cambios se pueden ver reflejados en la Ilustración 22-3.

Ilustración 22-3: Adición de elementos estéticos en el juego 3D.

Realizado por: Huilcamaygua, Cinthya, 2023.

Con respecto al primer punto donde era necesario tener dos estudiantes dentro de una misma aplicación, se tuvo que establecer que dentro de la aplicación de la habitación del laboratorio se encuentren dos brazos robóticos, uno para cada estudiante. Los estudiantes podrían mover el brazo y el otro estudiante podría ver ese movimiento, pero no podría interactuar con el brazo del otro estudiante. Adicional a esto, se creó una escena o scene de Unity para que ingresen los estudiantes, donde cada estudiante podría mover su brazo y no el brazo del compañero. Esta nueva escena agregada sería una interfaz de ingreso al laboratorio, como se puede observar en la Ilustración 23-3.

Ilustración 23-3: Ventana de ingreso de la aplicación de los estudiantes.

Realizado por: Huilcamaygua, Cinthya, 2023.

A continuación de este, aparecería el laboratorio, donde se hallarían dos brazos, y solo el menú correspondiente al brazo del estudiante seleccionado en un inicio, se encontraría activo. En la Ilustración 24-3, se puede observar que el menú del estudiante 1 se encuentra activo, mientras que no existe un menú para interactuar con el brazo del segundo estudiante.

Ilustración 24-3: Interfaz del estudiante 1

Realizado por: Huilcamaygua, Cinthya, 2023.

De igual forma, se debe presentar este hecho para el estudiante 2 que naturalmente puede ver activo su menú para interactuar con el brazo, mientras que el brazo del otro estudiante no presente ningún menú.

3.8.5 Desarrollo de la aplicación del profesor

La pantalla de inicio del profesor tiene un botón para entrar a la aplicación, sin la necesidad de ingresar credenciales para un inicio de sesión, tal como se observa en la Ilustración 25-3

Ilustración 25-3: Pasos para la conexión al servidor a través de un cliente SSH

Realizado por: Huilcamaygua, Cinthya, 2023.

La aplicación del profesor tiene el mismo laboratorio que el de los estudiantes, la diferencia es que el profesor no puede cambiar o alterar el movimiento de los brazos robóticos de los estudiantes. Solo contiene opciones para visualizar los movimientos que han guardado cada estudiante, detener el movimiento, y la función más importante es la de "Reiniciar intentos", pues, es la que rige si el estudiante puede o no realizar más movimientos en su aplicación (Ver Ilustración 26-3)

Ilustración 26-3: Vista del profesor en el laboratorio.

Realizado por: Huilcamaygua, Cinthya, 2023.

De manera que el panel de botones del brazo robótico sirve para visualizar movimientos anteriores y reiniciar el número de intentos para cada estudiante.

Asimismo, el profesor también puede ver los movimientos que se encuentre realizando el o los estudiantes si están conectados al mismo tiempo que el profesor. Los movimientos en tiempo real se verían reflejados en la pantalla de forma automática sin necesidad de presionar o ejecutar alguna función especial.

La programación para ambas aplicaciones tanto del profesor como del estudiante se encuentra en el entorno MQTT se encuentra en el ANEXO C.
La programación para ambas aplicaciones tanto del profesor como del estudiante se encuentra en el entorno HTTP se encuentra en el ANEXO D.

3.9 Funcionamiento del sistema

3.9.1 Utilizando el protocolo MQTT

Los elementos dentro del servidor AWS son el middleware o bróker EMQX, este bróker tiene una conexión directa con Node.js, a su vez este último se encuentra conectado directamente hacia la base de datos MongoDB. Node.js actúa como un cliente MQTT que se conecta a diferentes

tópicos y de acuerdo con cada tópico ejecuta funciones como guardar datos hacia la base de datos y acceder a los datos almacenados para enviarlos.

Por otra parte, se tiene a los clientes MQTT que son las aplicaciones creadas en Unity. Estos usuarios al ejecutar su aplicación envían un mensaje CONNECT para conectarse con el bróker (Ver Ilustración 27-3).

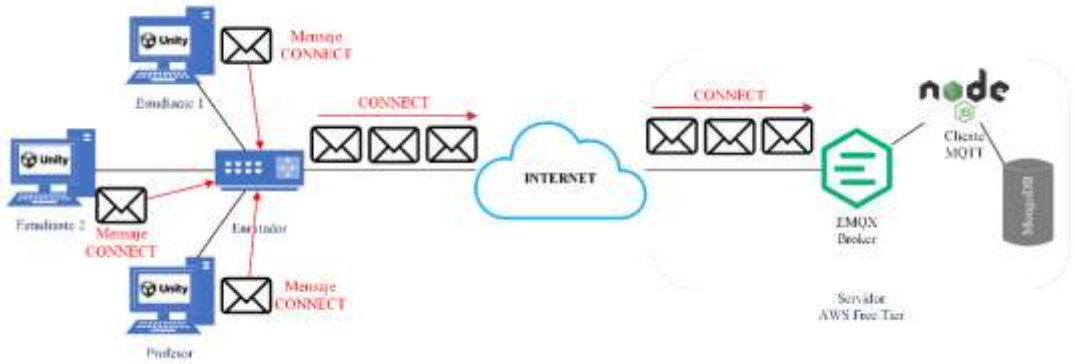


Ilustración 27-3: Mensaje CONNECT generado por las aplicaciones de RV para conectarse con el bróker.

Realizado por: Huilcamaygua, Cinthya, 2023.

El bróker atiende a ese paquete connect y se establece un conexión de manera exitosa, el bróker responde con un mensaje CONNACK (reconocimiento de conexión) para indicar si la conexión fue aceptada o si hay algún problema, como se puede ver en la Ilustración 28-3.

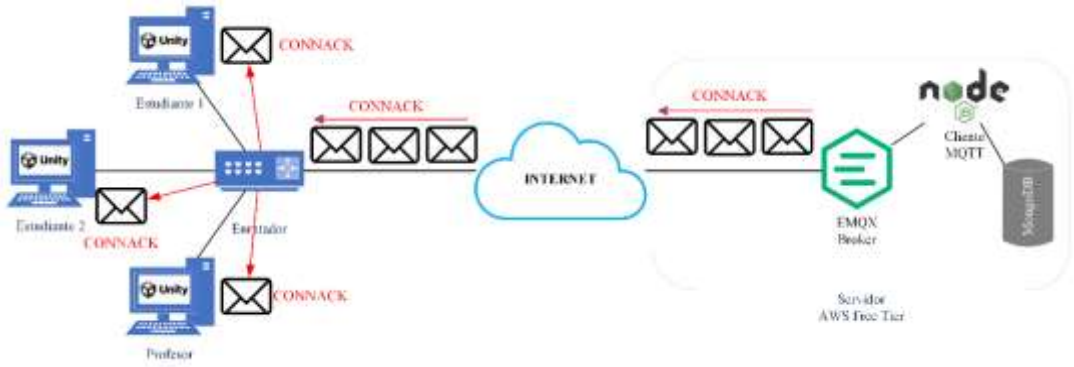


Ilustración 28-3: Mensaje CONNACK generado por el bróker.

Realizado por: Huilcamaygua, Cinthya, 2023.

Mediante esta conexión, el usuario logra mantener su enlace activo y por medio de paquetes de control transmitidos en intervalos regulares indican la permanencia activa en el sistema. Tras establecer esta conexión con el bróker, el usuario adquiere la capacidad de explorar el entorno virtual del laboratorio.

Si los usuarios estudiantes efectúan movimientos del brazo, estos movimientos se transmiten automáticamente hacia el bróker, el cual a su vez reenvía estos cambios a otros usuarios conectados, permitiéndoles percibir en tiempo real los movimientos realizados para cada ángulo del brazo. La aplicación de RV haría una publicación por cada vez que el estudiante interactúe con slider del brazo robótico, y el mensaje se enviaría como en la Ilustración 29-3.

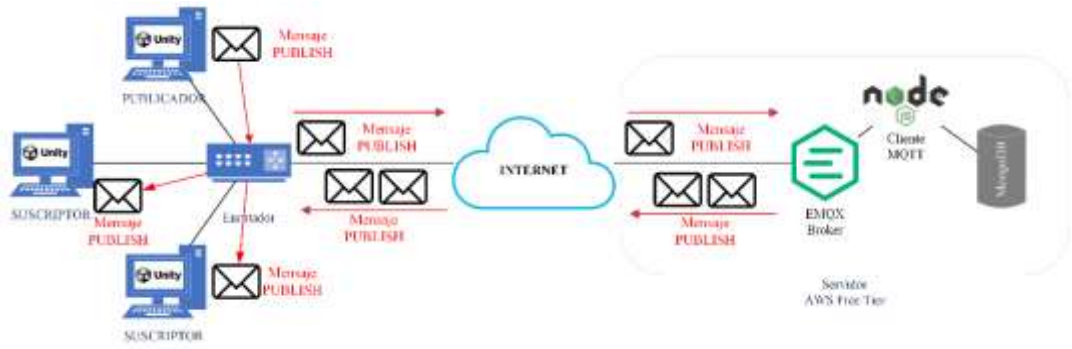


Ilustración 29-3: Mensaje PUBLISH desde un usuario hacia los clientes suscritos.

Realizado por: Huilcamaygua, Cinthya, 2023.

Si los estudiantes envían los intentos, el bróker recibe estos datos y como Node.js está actuando como un cliente MQTT que está esperando los datos de los intentos de los estudiantes, inmediatamente almacena esos datos en la base de datos. Este funcionamiento se repetiría para en caso de ambos estudiantes y también para una función que tiene el profesor que es la de reiniciar el número de intentos.

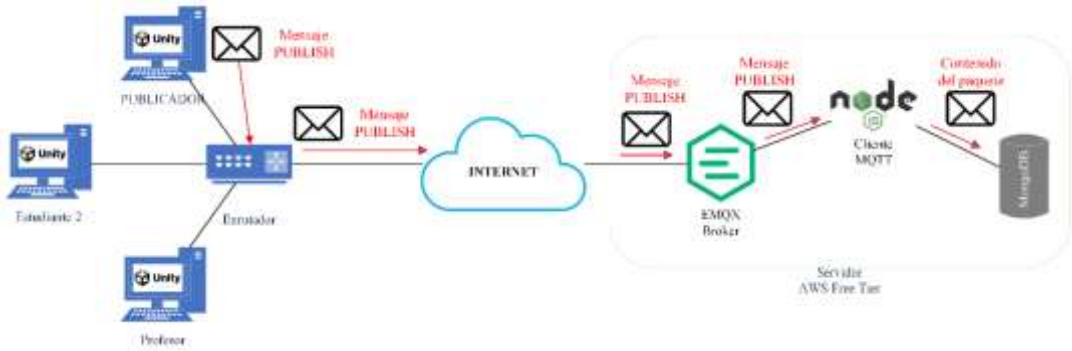


Ilustración 30-3: Mensaje PUBLISH desde un usuario publicador hacia la base de datos a través de Node.js.

Realizado por: Huilcamaygua, Cinthya, 2023.

En cuanto a la aplicación del profesor, su funcionamiento es distinto, ya que es él quien necesita solicitar o acceder a los datos almacenados en la base de datos. Por tanto, la justificación para utilizar Node.js radica en este aspecto. El profesor envía un mensaje dentro de un tópico específico y, Node.js esta suscrito a ese tópico y de acuerdo con el contenido del mensaje ejecuta una función

para acceder a la base de datos. Posteriormente, Node.js "retorna" una respuesta a través de un tópico al que el profesor está suscrito, llevando consigo los datos de los movimientos almacenados.

En la Ilustración 31-3, se observa cómo el profesor envía un mensaje en el tópico x. Dado que Node.js está suscrito a este tópico x, el mensaje llega y es leído por Node.js. Según el contenido del mensaje recibido, Node.js activa una función denominada "FindOne", que busca un dato en la base de datos. Este dato, transmitido como un mensaje de texto plano, se envía a través del tópico Y, al cual el profesor también está suscrito. De esta manera, el profesor puede obtener los movimientos almacenados.

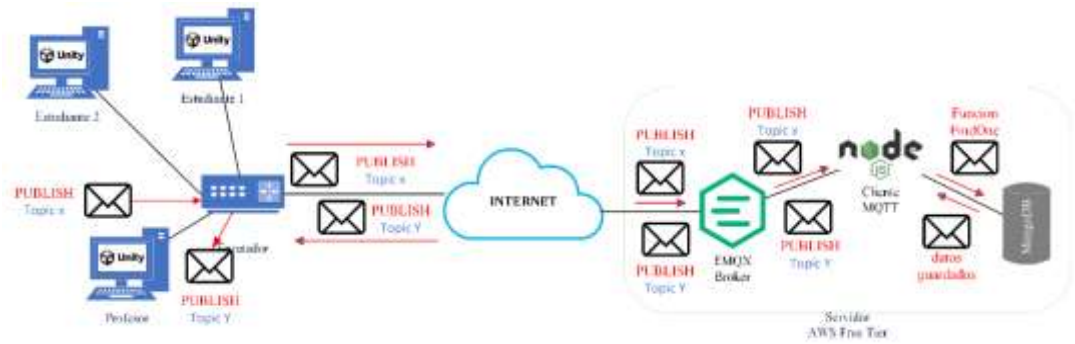


Ilustración 31-3: Obtención de los movimientos almacenados por parte del profesor.

Realizado por: Huilcamaygua, Cinthya, 2023.

La función que el profesor emplea para acceder a los datos almacenados también se encuentra disponible en el menú flotante de cada estudiante. Esto les permite visualizar los movimientos que han registrado y desean observar nuevamente lo realizado.

3.9.2 Escenario utilizando el protocolo HTTP

Al tener una topología de red centrada en servidor, se hace posible el uso de HTTP con los mismos elementos elegidos en el escenario de MQTT, exceptuando el uso del bróker y cediendo el trabajo de procesamiento o gestión de mensajes a node.js. Tal como se puede observar en la Ilustración 32-3.

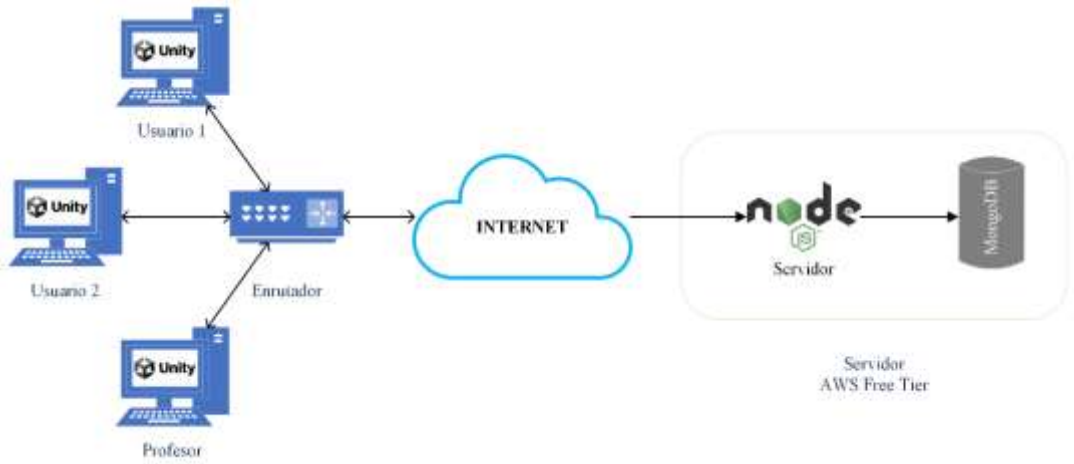


Ilustración 32-3: Escenario con el protocolo HTTP.

Realizado por: Huilcamaygua, Cinthya, 2023.

Considerando que el desarrollo de la aplicación ya fue realizado para el escenario MQTT, en esta sección se omite el desarrollo de un nuevo proyecto en 3D por lo que se ha duplicado el mismo proyecto finiquitado en una carpeta con un nombre diferente para distinguirlos.

Lo anterior no quiere decir que se ha usado el proyecto en su totalidad porque eso significaría utilizar el mismo protocolo de envío de datos, sino que se lo usó como una especie de plantilla en la que se ha basado para realizar el cambio de protocolo de comunicación. Ahora, como el proyecto que se modificó fue el duplicado del ya existente de mqtt, es normal que al principio los nombres de los scripts o de las funciones tengan relación con mqtt ya que ese fue el primer escenario que se desarrolló.

Por lo tanto, se procedió a realizar ciertos cambios en la programación. Se borraron las funciones de MQTT, se añadieron líneas de programación para crear métodos de Posteo, se ha empaquetado la información en formato JSON para el envío de datos y se ha sustituido las funciones de los botones de acción de la aplicación de RV.

Es importante resaltar que ciertas funciones ejecutadas por MQTT en relación con las interacciones en tiempo real no son factibles en este escenario. Esto se debe a que HTTP realiza solicitudes cada vez que requiere obtener o almacenar información, acciones que el escenario con HTTP llevaría a cabo mediante solicitudes GET y POST para la recuperación y almacenamiento de datos, porque HTTP no puede mantener una comunicación abierta para enviar o recibir los parámetros y solo se podría manejar con mensajes tipo GET y POST. Por lo que los usuarios profesor y estudiante 2 tendrían que refrescar o actualizar las variables de la aplicación con un F5 y actualizaría todas las variables dentro del juego de realidad virtual. De manera que, para ver el estado actual del brazo robótico, sería necesario ejecutar un refresco por cada vez que el estudiante

realice un movimiento o envíe un movimiento al servidor, pero este proceso sería incomodo para cualquiera de los dos usuarios que interactúen en tiempo real, dejando al escenario HTTP solamente con funciones de almacenado y revisión de datos.

3.9.3 Toma del número de muestras del sistema

En este trabajo de investigación, se ha configurado el sistema utilizando una selección de componentes que se han elegido específicamente por sus características o, dicho de otra forma, por métodos no probabilísticos. En este estudio, se tratan como variables independientes a los protocolos MQTT y HTTP, mientras que las variables dependientes comprenden los parámetros de rendimiento de la red, como el ancho de banda y tiempo de ida y vuelta

Dado que el sistema se encuentra en un entorno dinámico y la medición de estos parámetros puede variar en su frecuencia, se aborda la cuestión de cuántas veces medir estos parámetros para obtener una evaluación confiable. Si bien existen fórmulas o métodos probabilísticos, la determinación del número de ejecuciones requerido podría basarse en criterios estadísticos, como la fórmula para el cálculo del tamaño de muestra cuando se conoce la desviación estándar o el cálculo de la muestra finita e infinita, etc. (Hernández-Sampieri y Mendoza, 2018)

3.9.3.1 Análisis de potencia estadística

El análisis de potencia es una técnica utilizada para calcular la probabilidad de que una prueba estadística detecte un efecto significativo si realmente existe. Está enfocado en la planificación de experimentos que ayuda a determinar el tamaño de muestra necesario para evaluar la probabilidad de encontrar un efecto si este realmente está presente en la población. Esto evita la posibilidad de no encontrar efectos reales debido a un tamaño de muestra insuficiente.

La fórmula para calcular el tamaño de muestra (n) es la fórmula de prueba t de dos colas que depende de varios de factores, entre ellos, la desviación estándar. La desviación estándar es un valor que se puede calcular a partir de una muestra piloto, por lo tanto se utiliza la siguiente fórmula:

$$n = \left(\frac{(Z_{\alpha/2} + Z_{\beta}) \times \sigma}{D} \right)^2$$

Donde:

α: nivel de significancia.

β: probabilidad de cometer un error de Tipo II (no detectar una diferencia cuando realmente existe). 1 - β es la potencia de la prueba. Comúnmente, una potencia de 0.80 se considera aceptable.

σ: desviación estándar de las mediciones.

D: diferencia mínima que desea detectar.

Para calcular la desviación estándar de la muestra piloto, se ha seleccionado una de las gráficas más recurrentes en el experimento. Dado que se repite constantemente el mismo procedimiento, se optó por tomar datos de una de las tomas de una práctica de laboratorio. El cálculo de la desviación estándar da como resultado 1.1385. Los datos para obtener esta desviación estándar se encuentran en el ANEXO E.

De manera que se utiliza un valor de 0.6 para la D como diferencia de RTT a detectar en los experimentos. $Z_{\alpha/2}$ es el valor z correspondiente al nivel de confianza deseado para α= 0.5, $Z_{\alpha/2}$ es aproximadamente 1.96. Z_{β} es el valor z que corresponde al poder deseado. Para β =0.20 (potencia de 0.80) es aproximadamente 0.84.

$$n = \left(\frac{(1.96 + 0.84) \times 1.1385)}{0.6}\right)^2$$

$$n = \left(\frac{3.1878}{0.6}\right)^2$$

$$n = 28$$

Por lo que el número de muestras de acuerdo con esta fórmula será de 28 aunque para dividir el día de manera correcta y no tener decimales en la distribución de los minutos del día se usaría un total de 30 muestras, que por consiguiente se tomaría una muestra cada 48 minutos desde las 0 horas hasta las 24 horas.

Asimismo, también se tiene en cuenta que existe una regla empírica conocida como teorema del límite central. Este teorema sugiere que una muestra de alrededor de 30 observaciones puede proporcionar una base razonable para el análisis en trabajos exploratorios.

3.10 Técnicas e instrumentos de investigación

3.10.1 Técnicas de investigación

Elaboración: Esta técnica ha sido empleada para desarrollar los casos de estudio y los prototipos en Unity. Esto implica la creación detallada de los escenarios de prueba, la definición de las interacciones entre los dispositivos IoT y los elementos del entorno 3D, y la implementación de las funcionalidades específicas requeridas para la comunicación.

Síntesis: Se utiliza la técnica de síntesis para integrar la información recopilada de diferentes fuentes, como la revisión de literatura y los resultados de los casos de estudio. Se buscará identificar patrones, tendencias y relaciones entre los datos y las observaciones, con el objetivo de generar conclusiones y recomendaciones coherentes y comprensibles.

Analogía: Se ha aplicado la técnica de analogía para establecer comparaciones entre diferentes situaciones, contextos o tecnologías relacionadas.

3.10.2 Herramienta CloudWatch en AWS free tier

La herramienta CloudWatch es un servicio de monitoreo que permite supervisar recursos y aplicaciones en la nube de AWS. Este servicio se incluye en la capa gratuita o free tier. Es perfecto para ser utilizado dentro de este trabajo porque es necesario monitorear la utilización de los recursos de CPU y RAM de la instancia o máquina virtual del servidor centralizado.

La métrica de utilización de CPU si se encuentra por defecto en la instancia, mas no el del recurso utilización de RAM. Para ello se debe habilitar la recopilación de métricas para que se permita la lectura de uso de RAM.

3.10.2.1 Habilitar la recopilación de métricas

El primer paso es acceder al Servicio IAM o Administración de identidad y acceso, en la sección izquierda se encuentra un panel con varias opciones. Se elige la opción Roles, tal como se ve en la Ilustración 33-3.

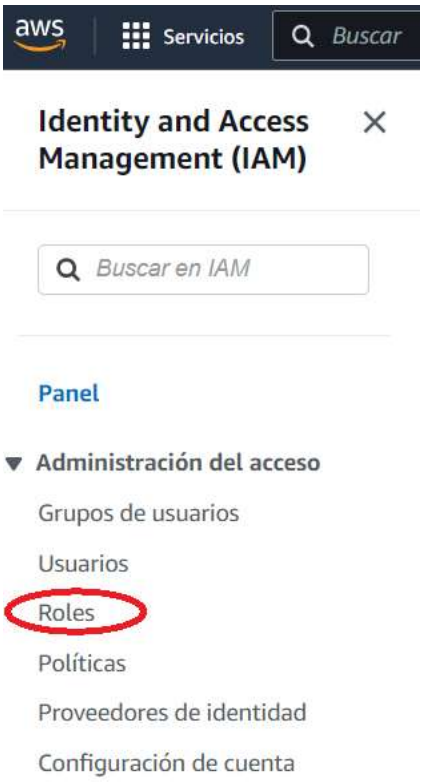


Ilustración 33-3: IAM para habilitar las métricas en CloudWatch

Fuente: https://us-east-1.console.aws.amazon.com/iamv2/home?region=us-east-1#/home

Enseguida aparece la sección de Roles donde se encuentran todos los roles existentes. Se da click en Crear rol (Ver Ilustración 34-3)

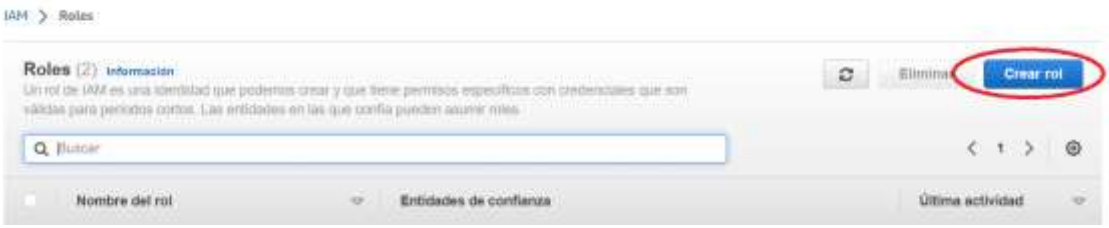


Ilustración 34-3: Ventana de Roles para la administración y creación de roles.

Fuente: https://us-east-1.console.aws.amazon.com/iamv2/home?region=us-east-1#/home

A continuación, se debe elegir la entidad y la instancia. En este caso, la entidad es Servicio de AWS, y la instancia es EC2 y continuar con el siguiente paso. (Ver Ilustración 35-3)

Ilustración 35-3: Ventana para la creación en la selección de la entidad.

Fuente: https://us-east-1.console.aws.amazon.com/iamv2/home?region=us-east-1#/home

Luego aparece el apartado de los permisos, en donde se debe buscar el nombre "CloudWatchAgentServerPolicy" que es el agente que permite compartir las métricas de la instancia EC2 con CloudWatch, tal como se puede apreciar en la Ilustración 36-3.

Ilustración 36-3: Ventana para la creación de roles en asignación de permisos.

Fuente: https://us-east-1.console.aws.amazon.com/iamv2/home?region=us-east-1#/home

Por último, se debe asignar un nombre al rol que se le ha colocado como Agente_cloudwatch (Ver Ilustración 37-3) y se finaliza la operación de la creación de roles.

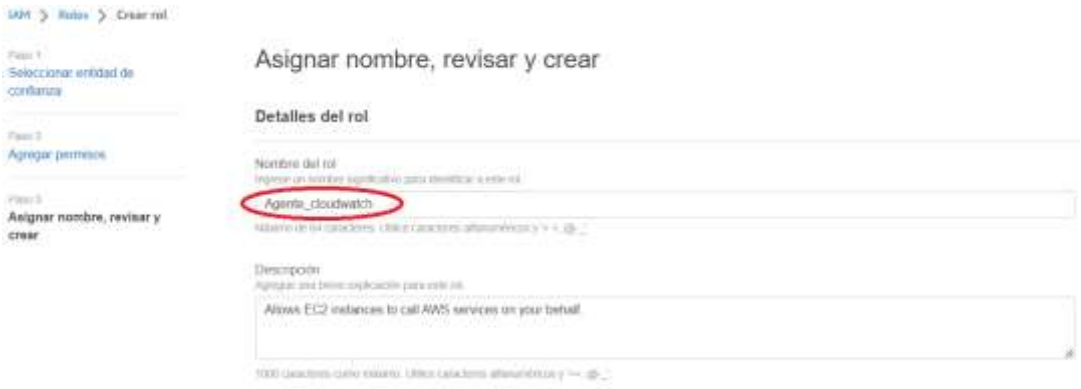


Ilustración 37-3: Ventana para la creación de roles en asignación de nombre.

Fuente: https://us-east-1.console.aws.amazon.com/iamv2/home?region=us-east-1#/home

3.10.2.2 Asignación de rol a la instancia

Después de crear el rol, se lo debe asignar a la instancia a la cual se quiere acceder la lectura de las métricas. Para ello se debe seleccionar la instancia, dirigirse a la pestaña de acciones, luego a seguridad y allí está la opción Modificar rol de IAM, tal como se ve en la Ilustración 38-3.

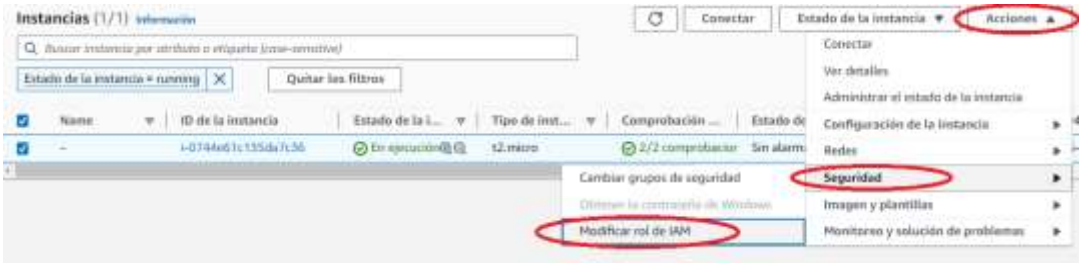


Ilustración 38-3: Modificación de rol de la instancia.

Fuente: https://us-east-1.console.aws.amazon.com/ec2/home

Esta opción seleccionada redirige a otra ventana en donde se debe escoger el rol que se ha creado anteriormente para asociarlo a la instancia, y se finaliza con el botón “Actualizar rol de IAM” (Ver Ilustración 39-3).

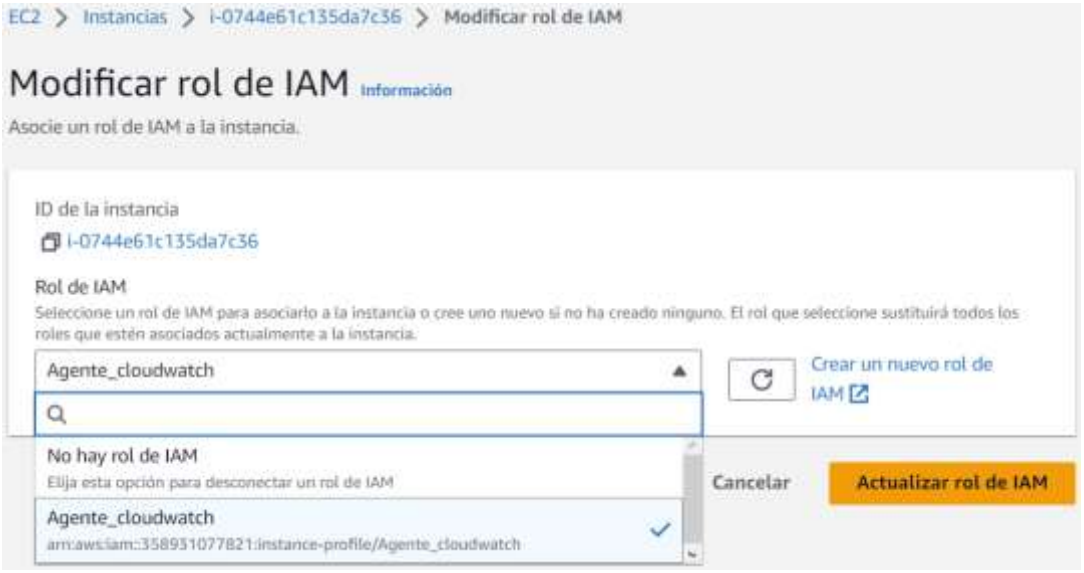


Ilustración 39-3: Asociación de rol a la instancia EC2.

Fuente: https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances:instanceState=running

3.10.2.3 Instalación de CloudWatch

Este servicio no suele estar instalado en la instancia. La descarga de CloudWatch se lo puede hacer a través de cualquier otra aplicación que permita la conexión por SSH. Allí se debe colocar el comando para la descarga de CloudWatch que es:

wget https://s3.amazonaws.com/amazoncloudwatch-agent/ubuntu/amd64/latest/amazon-cloudwatch-agent.deb

Después de la descarga se debe abrir el paquete con el comando: sudo dpkg -i -E ./amazon-cloudwatch-agent.deb

Luego se escribe el comando: sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-config-wizard. Esto ejecuta una especie de instalador donde se configuran varias opciones. (Ver Ilustración 40-3).

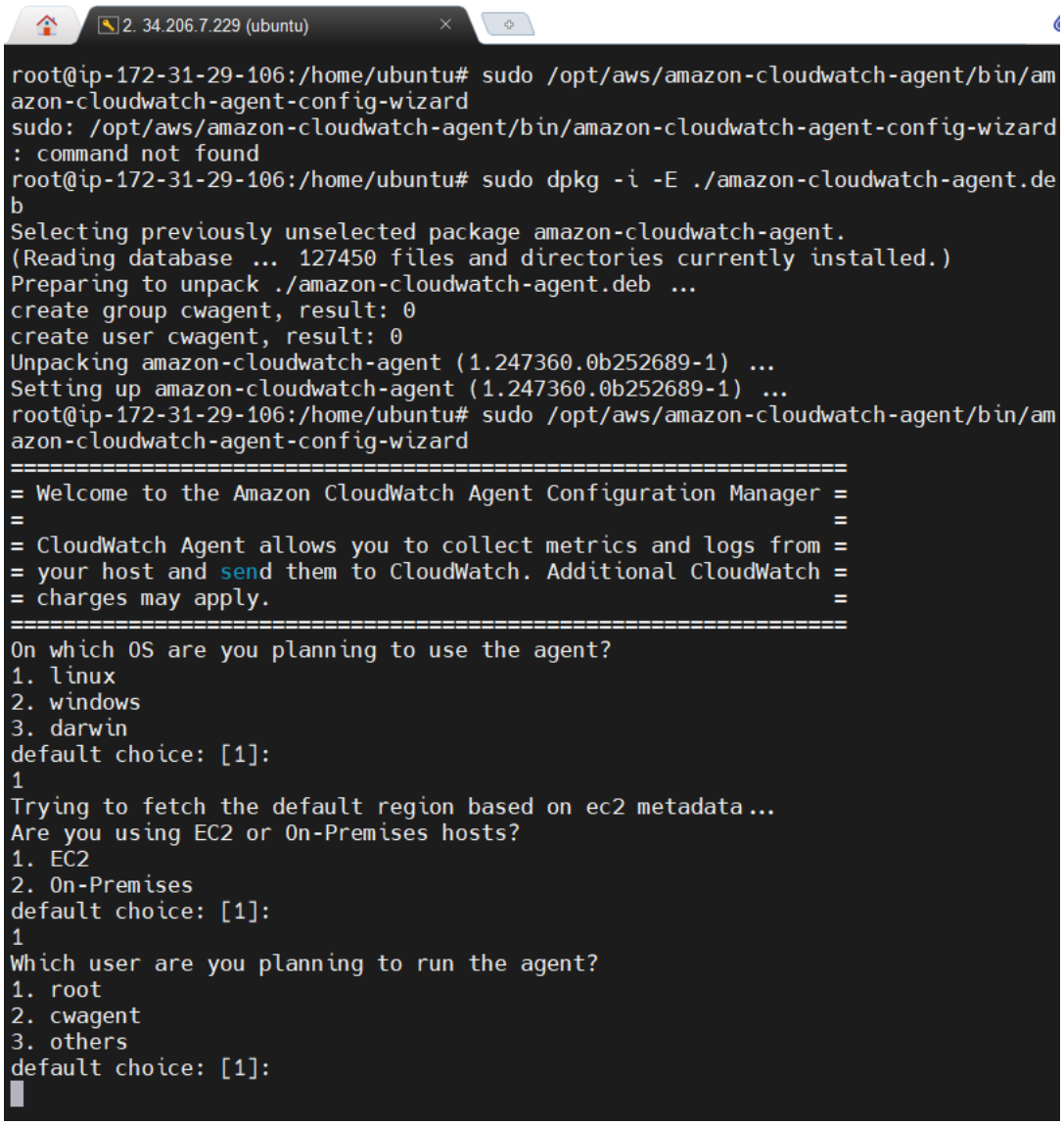


Ilustración 40-3: Instalación de CloudWatch

Para conocer si el agente CloudWatch se encuentra en estado “running” se coloca el comando amazon-cloudwatch-agent-ctl -a status, tal como se ve en la ilustración 41-3.

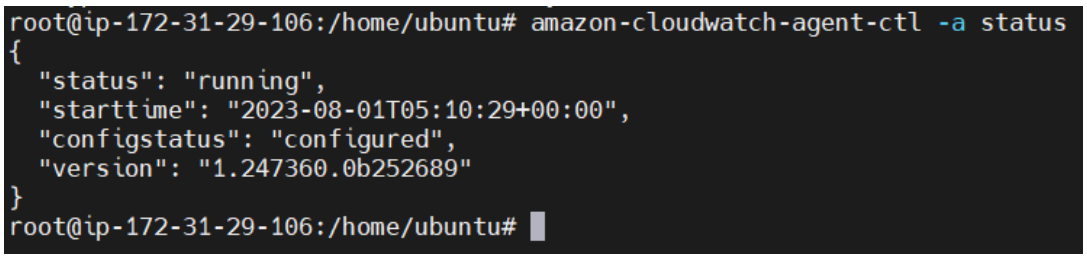


Ilustración 41-3: Estado de CloudWatch en running.

3.10.2.4 Uso de CloudWatch en AWS

Una vez realizada la instalación de este servicio a través de la consola, ya se puede acceder desde la página de AWS con el nombre de CloudWatch. Una vez allí, en el panel de lado izquierdo está el apartado de Métricas de donde se va a seleccionar la opción Todas las métricas. En el panel aparece una sección llamado CWAgente que contiene 19 métricas en él. (Ver Ilustración 42-3)

Ilustración 42-3: Nuevo sector de métricas añadidos en la creación de CWAgent

Ese nuevo espacio de CWAgent de19 métricas se divide en 2 secciones más. En la Ilustración 43-3se muestra una opción que contiene una métrica y otra opción que contiene 18 métricas.

Ilustración 43-3: Nuevo sector de métricas añadidos en la creación de CWAgent

En la opción que tiene una sola métrica, se encuentra la métrica de uso de la memoria en forma de porcentaje. En la Ilustración 44-3, se muestra como mem_used_percent y es la que se ha ocupado para poder generar una gráfica del porcentaje de uso de la memoria de la instancia EC2.

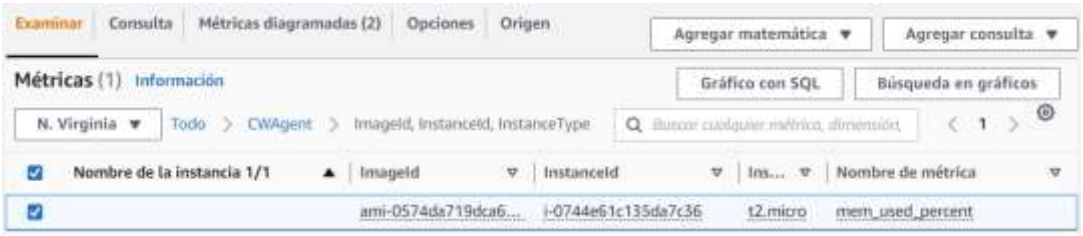


Ilustración 44-3: Métrica porcentaje de uso de la memoria.

Mientras que el sector de 18 métricas contiene una métrica importante que sirve para visualizar el porcentaje de uso del disco de la instancia para poder conocer cuanto espacio esta libre. En la Ilustración 45-3 se puede ver que la métrica seleccionada corresponde a la memoria principal de la instancia EC2 como dispositivo xvda15.

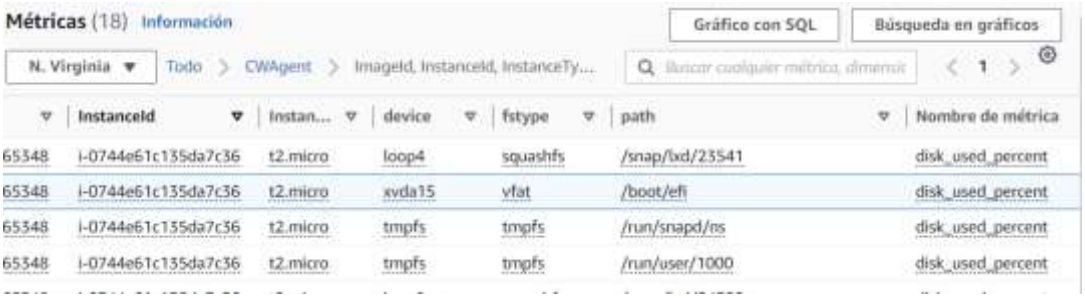


Ilustración 45-3: Métrica porcentaje de uso del disco

Fuente: https://us-east-1.console.aws.amazon.com/cloudwatch/home

Finalmente, para agregar esas métricas en formas de grafico se lo hizo seleccionando la pestaña Acciones y en la opción Añadir al panel se puede anexar el grafico hacia el panel principal de CloudWatch, tal como se puede ver en la Ilustración 46-3.

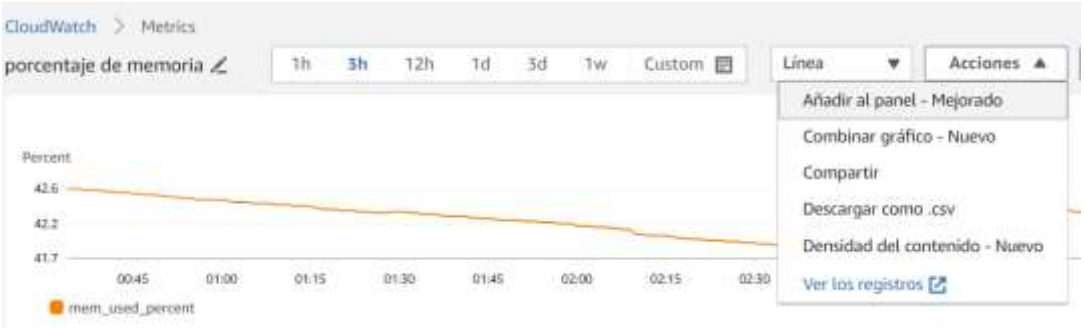


Ilustración 46-3: Función añadir grafico de métrica al panel principal de CloudWatch

Fuente: https://us-east-1.console.aws.amazon.com/cloudwatch/home

De manera que se agrega a un panel que haya sido creado previamente o se puede crear uno nuevo, también se puede seleccionar el tipo de gráfica, que en este caso es tipo Línea y el título

del grafico se añade a criterio del investigador, en este caso es Porcentaje de memoria (Ver Ilustración 47-3)

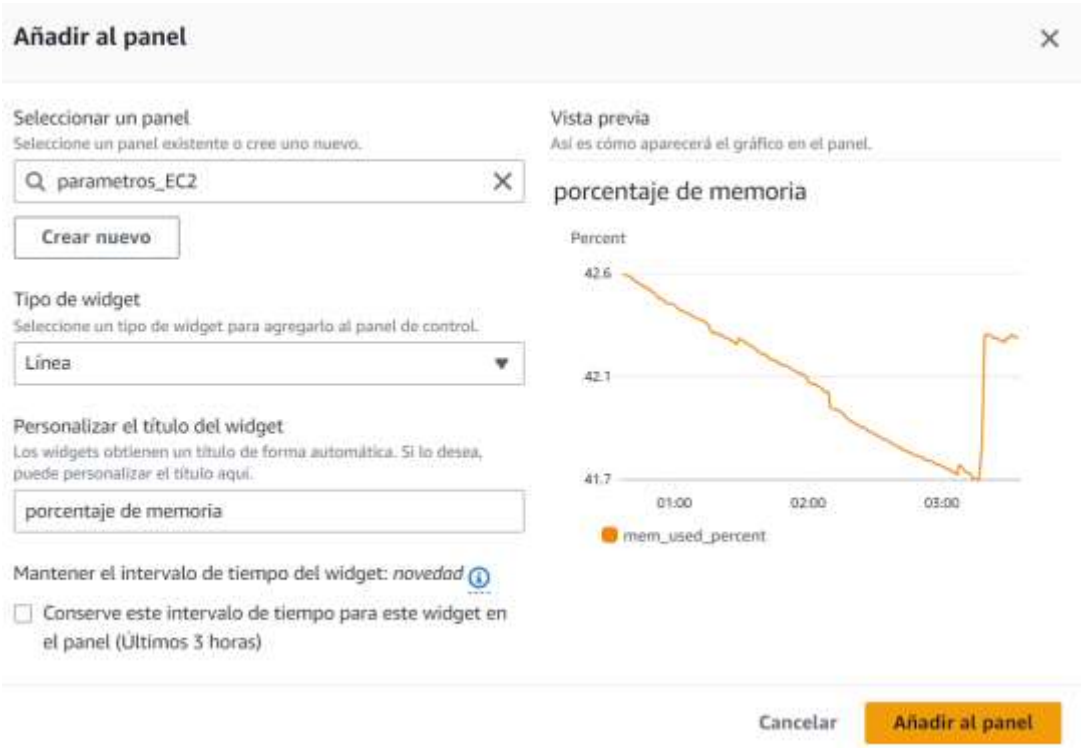


Ilustración 47-3: Ventana para añadir grafico de la métrica mem_used_percent al panel principal de CloudWatch

Fuente: https://us-east-1.console.aws.amazon.com/cloudwatch/home

Este procedimiento se realiza para todas las métricas que se quieren visualizar en tiempo real. En el contexto especifico de esta investigación, se optó por revisar las métricas:

Porcentaje de uso de memoria RAM (Ver Ilustración 48-3)

- Porcentaje de uso de CPU (Ver Ilustración 48-3)
- Porcentaje de uso del disco o almacenamiento (solo visualización)

Ilustración 48-3: Gráficos añadidos al panel principal de CloudWatch

Fuente: https://us-east-1.console.aws.amazon.com/cloudwatch/home

3.10.3 Herramienta Wireshark

Wireshark es una poderosa herramienta de código abierto utilizada para el análisis de redes. También conocida como "sniffer de red", permite capturar y examinar el tráfico de red en tiempo real, proporcionando una visión detallada de la comunicación que ocurre en una red.

Permite a los usuarios examinar el contenido de estos paquetes y revelar información valiosa, como direcciones IP, puertos, protocolos utilizados y datos transmitidos. Al identificar patrones y anomalías en el tráfico de red, Wireshark se convierte en una herramienta esencial para diagnosticar problemas de conectividad, rendimiento y seguridad en una red. (Ver Ilustración 49-3)

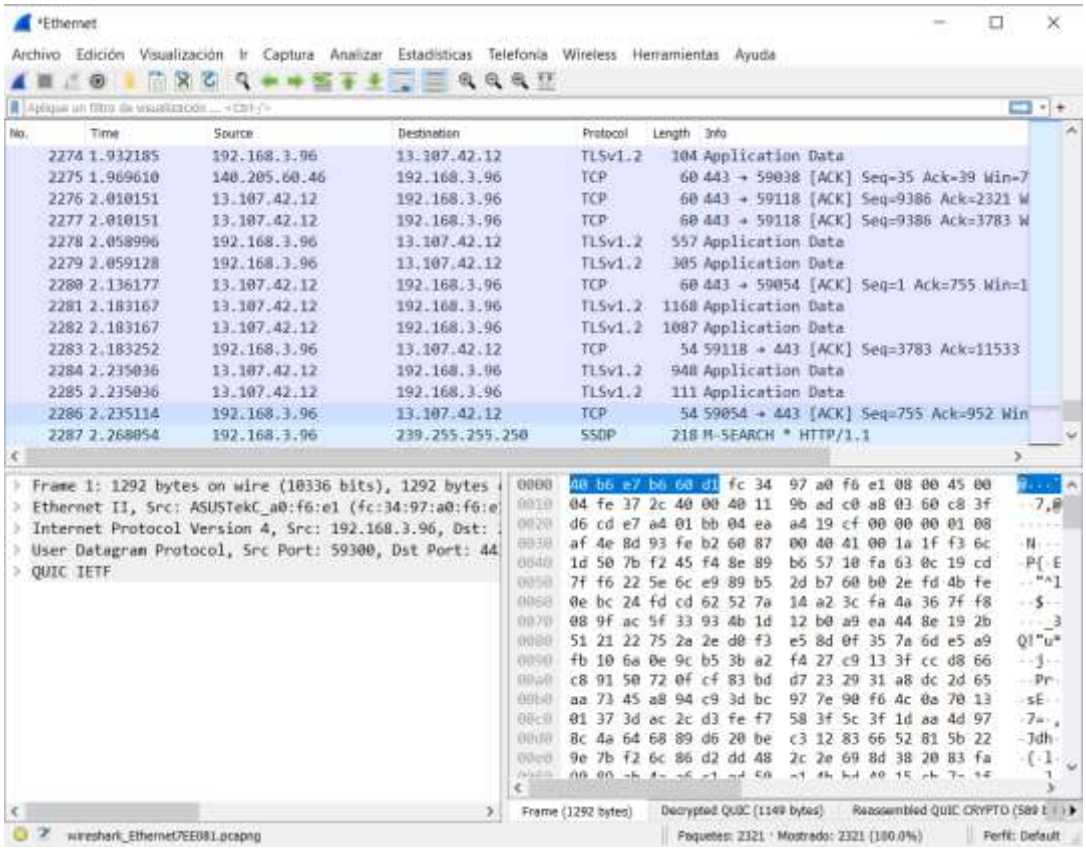


Ilustración 49-3: Captura de tráfico con Wireshark.

Realizado por: Huilcamaygua, Cinthya, 2023.

Ahora, si bien puede examinar el tráfico de la red, se necesita de la utilización de filtros para obtener datos específicos como lo pueden ser ciertos protocolos o direcciones de ip, en fin, existen muchas formas de filtrar un tráfico dependiendo de la información que se necesite.

3.10.3.1 Utilización de Wireshark

El primer paso para obtener paquetes específicos es estableciendo un filtro. Para este trabajo de investigación se ha establecido capturar solamente el tráfico que involucra al usuario y al servidor. Esto es cualquier otro tráfico debe quedar fuera de este contexto. Por lo tanto, el filtro sería el siguiente:

((ip.src == 34.206.7.229) or (ip.dst == 34.206.7.229)) and not (tcp.port == 22 or udp.port == 22)

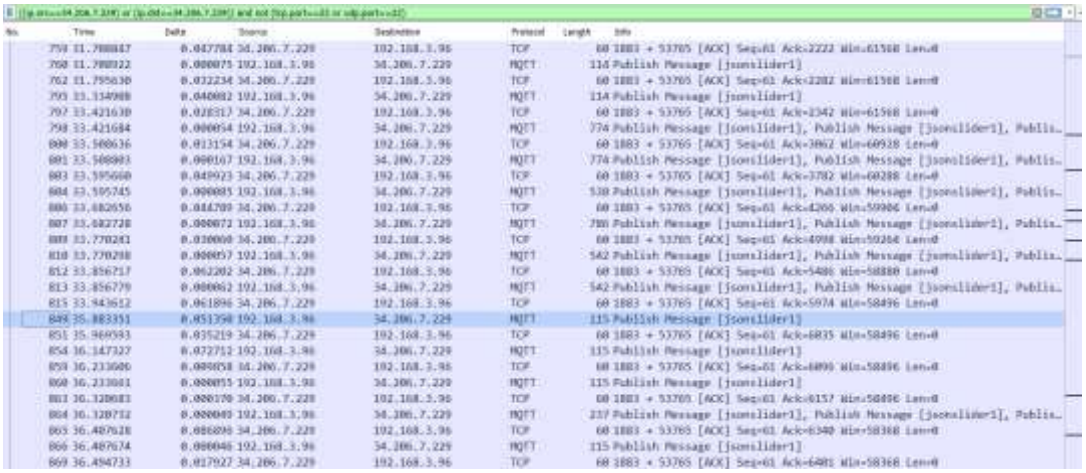


Ilustración 50-3: Captura de tráfico a través de Wireshark con filtro de una prueba piloto.

Realizado por: Huilcamaygua, Cinthya, 2023.

En la Ilustración 50-3, se observa que el filtro busca paquetes que involucran la dirección IP 34.206.7.229 como dirección de origen o destino, pero excluye aquellos que utilizan el puerto 22 para TCP o UDP. Esto es útil para observar tráfico que involucre la dirección IP del servidor, pero no esté relacionado con conexiones SSH.

El siguiente paso es obtener los parámetros de rendimiento de la red. Con la captura de tráfico filtrada se obtienen las gráficas e información el RTT, ancho de banda utilizado y pérdida de paquetes.

El ancho de banda utilizado o tasa de transferencia en toda la comunicación se lo puede obtener desde la opción “Conversaciones” ubicada en la pestaña de Estadísticas. Aquí se encuentra la información del ancho de banda upstream (subida) y downstream (bajada) a modo de tabla resumen dentro de la ventana Ethernet.

Para determinar el tiempo de ida y vuelta en esta captura de tráfico, se lo realiza desde la ventana de conversaciones y se selecciona la opción Gráfica. Allí aparece una nueva ventana en donde hay que seleccionar el tipo de gráfica y se escoge Round Trip Time (Tiempo de ida y vuelta). Asimismo, existe una gráfica para el tráfico desde el origen (usuario) - destino (servidor) y viceversa, tan solo seleccionando la opción Cambiar dirección.

Los protocolos MQTT y HTTP se envían sobre TCP, por lo que existe garantía de que la información llegue al servidor. Pero también pueden ocurrir errores en la transmisión de datos como errores TCP ocurridas durante la ejecución del escenario de red y tengan que ser retransmitidas. Por tanto, en Wireshark es posible hallar este tipo de errores en la ventana de Estadísticas de E/S donde los errores se destacan por ser barras de color rojo.

CAPÍTULO IV

4. MARCO DE ANÁLISIS E INTERPRETACIÓN DE RESULTADOS

Tras la implementación de los componentes de la red, se han recopilado datos cruciales sobre el desempeño del sistema que brinda acceso a los servicios de realidad virtual. Esto ha involucrado el uso de herramientas para la obtención de datos del rendimiento del sistema como Wireshark. que captura el tráfico que pasa por un puerto. Los indicadores de rendimiento de la red abordaron aspectos como el tiempo de ida y vuelta (RTT), la pérdida de paquetes y el ancho de banda utilizado, así mismo también se pueden medir el uso de los recursos de RAM y CPU del servidor con ayuda de CloudWatch.

La obtención de los parámetros de rendimiento de la red se logra al capturar el flujo completo de tráfico generado por los usuarios que están conectados, pero de manera aislada. Hay que tomar en cuenta que la captura del tráfico se lo debe realizar en un dispositivo diferente para cada usuario porque si ejecutamos todas las aplicaciones en un mismo computador, no es posible diferenciar el tráfico generado por cada usuario.

Cada usuario ejecuta diversas actividades dentro de su entorno virtual, y se analizarían para diferentes casos, cuando se conecta un solo estudiante, cuando se conecta solamente el profesor, cuando se conecta el profesor y estudiante al mismo tiempo, y por último el escenario más crítico, que es cuando todos los usuarios están conectados simultáneamente. La toma de las muestras se lo ha realizado bajo las mismas condiciones para cada caso.

El presente trabajo está dirigido para simular un laboratorio sin la necesidad de acudir de forma presencial. Este enfoque permite la realización de prácticas y clases en un entorno virtual. Se han definido criterios para evaluar escenarios, tanto críticos como no críticos. Uno de los factores clave en esta evaluación es el tipo de conexión a internet. La conexión a internet se analiza en términos del plan más elemental disponible para cualquier usuario. Este criterio abarca el escenario más crítico, basado en un plan de internet mínimo. Si bien la mejora de parámetros de rendimiento podría lograrse con planes de internet más avanzados, estos representarían situaciones favorables con las que pocos podrían relacionarse. En la actualidad, el plan de internet residencial más básico ofrece una velocidad de 20 Mbps que es el plan que se ha utilizado para realizar las pruebas.

Un factor adicional de importancia es la saturación del canal o de la conexión a internet. Esta situación puede fluctuar según la cantidad de usuarios conectados y la cantidad de datos que estén

siendo transmitidos. Analizar el tráfico en diferentes momentos del día puede revelar escenarios ideales, cuando el canal está desocupado, y escenarios desfavorables, cuando el canal está congestionado. Esta evaluación abarcaría tanto el rendimiento de la conexión local como la conexión hasta el proveedor de servicios de internet (ISP), pues el proveedor comparte un mismo canal de transmisión entre diferentes clientes o abonados por lo que a lo largo del día ese canal compartido también se puede saturar hasta llegar a la ISP.

Al analizar previamente que un número de muestras adecuado para el presente trabajo es de 28 según la fórmula de cálculo de la muestra con desviación estándar conocida, se ha ajustado este número al de la regla empírica de 30 observaciones para este análisis para los escenarios usando tanto MQTT como HTTP.

4.1 Análisis de paquetes con Wireshark

4.1.1 En el escenario MQTT

Apenas el usuario se conecta a través de la aplicación de RV, ésta envía información hacia el servidor para mantener una conexión. Todos los paquetes que intervienen en la conexión como los paquetes CONNECT, ACKs para la sincronización de datos y la suscripción a diferentes tópicos tienen diferentes tamaños de paquetes o paquetes SUSCRIBE. Por ejemplo, los mensajes de sincronización tienen un tamaño de paquete de 66 bytes, para el comando de conexión hacia el bróker el paquete pesa 87 bytes, la suscripción a distintos tópicos varían en función de los parámetros necesitados como la suscripción al tópico llamado “reset” que tiene un tamaño de paquete de 148 bytes, todos los demás ACKs tiene un peso de 54 y las suscripciones tienen un peso 60 bytes. En la columna cumulative bytes de Wireshark se puede ver que durante toda la sincronización de datos para la conexión hacia el servidor, la suma de todos esos datos es de 1194 y lo realiza en 2.21 segundos (Ver Ilustración 1-4)

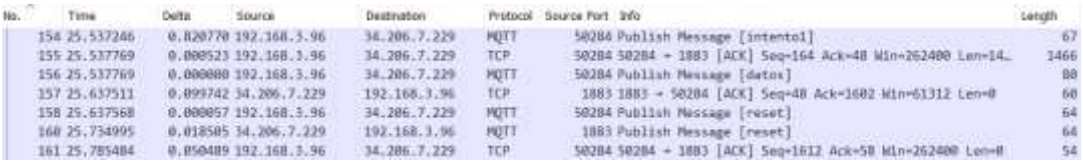
No.	Time	Delta	Source	Destination	Protocol	Source Port	Info	Length	cumulative bytes
66	7.226678	0.401589	192.168.3.96	34.206.7.229	TCP	50284	50284 → 1883 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 W…	66	66
68	7.324649	0.029288	34.206.7.229	192.168.3.96	TCP	1883	1883 → 50284 [SYN, ACK] Seq=0 Ack=1 Win=62727 Len=0…	66	132
69	7.324830	0.000181	192.168.3.96	34.206.7.229	TCP	50284	50284 → 1883 [ACK] Seq=1 Ack=1 Win=262400 Len=0	54	186
70	7.328685	0.003855	192.168.3.96	34.206.7.229	MQTT	50284	Connect Command	87	273
71	7.425638	0.096953	34.206.7.229	192.168.3.96	TCP	1883	1883 → 50284 [ACK] Seq=1 Ack=34 Win=62720 Len=0	60	333
72	7.426231	0.000593	34.206.7.229	192.168.3.96	MQTT	1883	Connect Ack	60	393
73	7.432983	0.006752	192.168.3.96	34.206.7.229	MQTT	50284	Subscribe Request (id=1) [#]	62	455
74	7.531151	0.098168	34.206.7.229	192.168.3.96	MQTT	1883	Subscribe Ack (id=1)	60	515
75	7.578127	0.046976	192.168.3.96	34.206.7.229	TCP	50284	50284 → 1883 [ACK] Seq=42 Ack=10 Win=262400 Len=0	54	569
88	9.239876	0.411663	192.168.3.96	34.206.7.229	MQTT	50284	Subscribe Request (id=2) [Intentos]	69	638
90	9.338085	0.017953	34.206.7.229	192.168.3.96	MQTT	1883	Subscribe Ack (id=2)	60	698
91	9.338188	0.000103	192.168.3.96	34.206.7.229	MQTT	50284	Subscribe Request (id=3) [reset], Subscribe Request…	148	846
92	9.436505	0.098317	34.206.7.229	192.168.3.96	MQTT	1883	Subscribe Ack (id=3)	60	906
93	9.436613	0.000108	34.206.7.229	192.168.3.96	MQTT	1883	Subscribe Ack (id=4)	60	966
94	9.436613	0.000000	34.206.7.229	192.168.3.96	MQTT	1883	Subscribe Ack (id=5)	60	1026
95	9.436642	0.000029	192.168.3.96	34.206.7.229	TCP	50284	50284 → 1883 [ACK] Seq=151 Ack=30 Win=262400 Len=0	54	1080
96	9.436663	0.000021	34.206.7.229	192.168.3.96	MQTT	1883	Subscribe Ack (id=6)	60	1140
97	9.436684	0.000021	192.168.3.96	34.206.7.229	TCP	50284	50284 → 1883 [ACK] Seq=151 Ack=35 Win=262400 Len=0	54	1194

Ilustración 1-4: Captura de paquetes del inicio de la aplicación de realidad virtual como cliente MQTT.

Realizado por: Huilcamaygua, Cinthya, 2023.

Estas operaciones al iniciar la aplicación se realizan siempre, teniendo el mismo valor de datos acumulados de 1194 bytes. Esto se afirma porque en las diferentes ejecuciones del sistema, estos paquetes y este valor se mantiene igual en todas las pruebas.

Cuando el estudiante realiza el envío de un intento de la práctica de laboratorio, se envía un mensaje PUBLISH que tiene un peso de 67 bytes, y a continuación se envía el mensaje que quiere publicar bajo el tópico intento1, que tiene un tamaño de 1466 bytes. Asimismo, también se actualizan otros tópicos cuando se envían los datos del intento 1. Como se ve en la Ilustración 2-4.

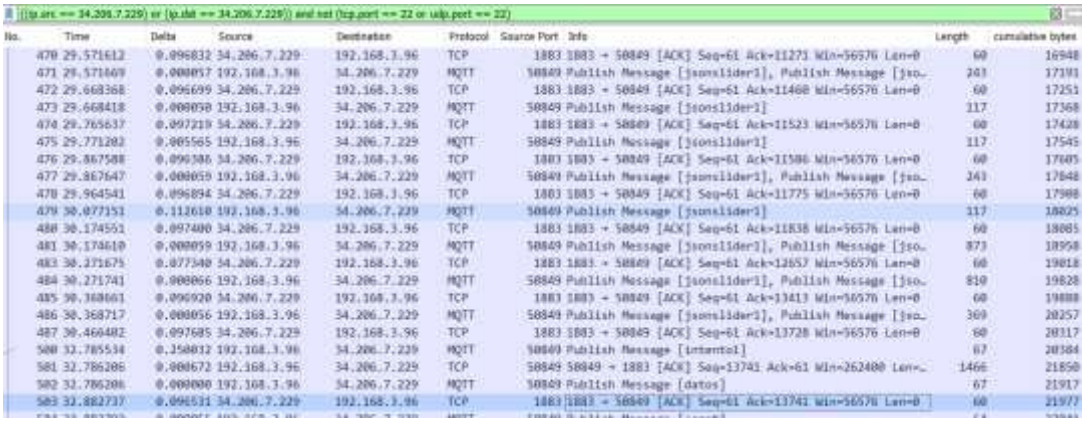


No.	Time	Delta	Source	Destination	Protocol	Source Port	Info	Length
154	25.537246	0.820770	192.168.3.96	34.206.7.229	MQTT	50284	Publish Message [intento1]	67
155	25.537769	0.000523	192.168.3.96	34.206.7.229	TCP	50284	50284 → 1883 [ACK] Seq=164 Ack=48 Win=262400 Len=14…	1466
156	25.537769	0.000000	192.168.3.96	34.206.7.229	MQTT	50284	Publish Message [datos]	80
157	25.637511	0.099742	34.206.7.229	192.168.3.96	TCP	1883	1883 → 50284 [ACK] Seq=48 Ack=1602 Win=61312 Len=0	60
158	25.637568	0.000057	192.168.3.96	34.206.7.229	MQTT	50284	Publish Message [reset]	64
160	25.734995	0.018585	34.206.7.229	192.168.3.96	MQTT	1883	Publish Message [reset]	64
161	25.785484	0.050489	192.168.3.96	34.206.7.229	TCP	50284	50284 → 1883 [ACK] Seq=1612 Ack=58 Win=262400 Len=0	54

Ilustración 2-4: Captura de paquetes cuando el estudiante envía un intento de la práctica de laboratorio.

Realizado por: Huilcamaygua, Cinthya, 2023.

Cuando el estudiante mueve el brazo robótico, el valor de los ángulos toman diferentes tamaños de paquete que van desde los 116 bytes hasta los 940 bytes. Y cada vez que el estudiante envía un intento, el paquete siempre es de 1466 bytes. (Ver ilustración 3-4)



No.	Time	Delta	Source	Destination	Protocol	Source Port	Info	Length	cumulative bytes
470	29.571612	0.096832	34.206.7.229	192.168.3.96	TCP	1883	1883 → 50849 [ACK] Seq=61 Ack=11271 Win=56576 Len=0	60	16948
471	29.571669	0.000057	192.168.3.96	34.206.7.229	MQTT	50849	Publish Message [jsonslider1], Publish Message [jso…	243	17191
472	29.668368	0.096699	34.206.7.229	192.168.3.96	TCP	1883	1883 → 50849 [ACK] Seq=61 Ack=11460 Win=56576 Len=0	60	17251
473	29.668418	0.000050	192.168.3.96	34.206.7.229	MQTT	50849	Publish Message [jsonslider1]	117	17368
474	29.765637	0.097219	34.206.7.229	192.168.3.96	TCP	1883	1883 → 50849 [ACK] Seq=61 Ack=11523 Win=56576 Len=0	60	17428
475	29.771202	0.005565	192.168.3.96	34.206.7.229	MQTT	50849	Publish Message [jsonslider1]	117	17545
476	29.867588	0.096386	34.206.7.229	192.168.3.96	TCP	1883	1883 → 50849 [ACK] Seq=61 Ack=11586 Win=56576 Len=0	60	17605
477	29.867647	0.000059	192.168.3.96	34.206.7.229	MQTT	50849	Publish Message [jsonslider1], Publish Message [jso…	243	17848
478	29.964541	0.096894	34.206.7.229	192.168.3.96	TCP	1883	1883 → 50849 [ACK] Seq=61 Ack=11775 Win=56576 Len=0	60	17908
479	30.077151	0.112610	192.168.3.96	34.206.7.229	MQTT	50849	Publish Message [jsonslider1]	117	18025
480	30.174551	0.097400	34.206.7.229	192.168.3.96	TCP	1883	1883 → 50849 [ACK] Seq=61 Ack=11838 Win=56576 Len=0	60	18085
481	30.174610	0.000059	192.168.3.96	34.206.7.229	MQTT	50849	Publish Message [jsonslider1], Publish Message [jso…	873	18958
483	30.271675	0.077340	34.206.7.229	192.168.3.96	TCP	1883	1883 → 50849 [ACK] Seq=61 Ack=12657 Win=56576 Len=0	60	19018
484	30.271741	0.000066	192.168.3.96	34.206.7.229	MQTT	50849	Publish Message [jsonslider1], Publish Message [jso…	810	19828
485	30.368661	0.096920	34.206.7.229	192.168.3.96	TCP	1883	1883 → 50849 [ACK] Seq=61 Ack=13413 Win=56576 Len=0	60	19888
486	30.368717	0.000056	192.168.3.96	34.206.7.229	MQTT	50849	Publish Message [jsonslider1], Publish Message [jso…	369	20257
487	30.466482	0.097685	34.206.7.229	192.168.3.96	TCP	1883	1883 → 50849 [ACK] Seq=61 Ack=13728 Win=56576 Len=0	60	20317
500	32.785534	0.350812	192.168.3.96	34.206.7.229	MQTT	50849	Publish Message [intento1]	67	20384
501	32.786206	0.000672	192.168.3.96	34.206.7.229	TCP	50849	50849 → 1883 [ACK] Seq=13741 Ack=61 Win=262400 Len=…	1466	21850
502	32.786206	0.000000	192.168.3.96	34.206.7.229	MQTT	50849	Publish Message [datos]	67	21917
503	32.882737	0.096531	34.206.7.229	192.168.3.96	TCP	1883	1883 → 50849 [ACK] Seq=61 Ack=13741 Win=56576 Len=0	60	21977

Ilustración 3-4: Captura de paquetes cuando el estudiante mueve el brazo robótico.

Realizado por: Huilcamaygua, Cinthya, 2023.

4.1.2 En el escenario HTTP

Como el funcionamiento de HTTP es distinto al de MQTT, se establecieron algunas funciones para que al iniciar la aplicación se obtengan los datos, para lo cual se realizaron solicitudes GET para obtener los datos como el número de intentos y los intentos almacenados si es que existe alguno guardado.

Además, la naturaleza de HTTP es de establecer una nueva conexión por cada vez que se realice una solicitud, es decir, por cada solicitud POST o GET iniciaría una nueva comunicación. En la Ilustración 4-4 se puede ver la captura de Wireshark para establecer la primera conexión con el servidor, pues realiza una solicitud GET para inicializar las variables de la aplicación de RV. El inicializar todas las variables le lleva alrededor de 1081 bytes realizar esa sincronización, y lo realiza en un lapso de 5.21 segundos

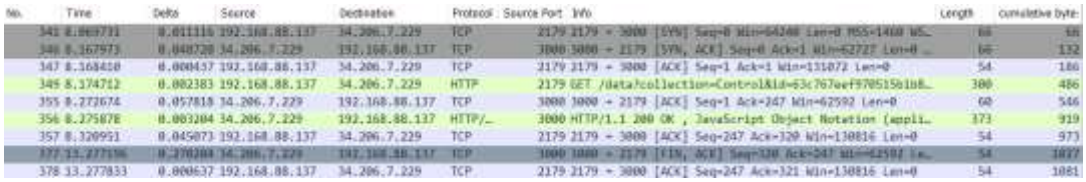


Ilustración 4-4: Captura de paquetes al iniciar la aplicación del estudiante.

Realizado por: Huilcamaygua, Cinthya, 2023.

Cuando el estudiante envía el valor del intento presionando el botón de enviar intento, la aplicación envía solicitudes de tipo POST en la ruta /control y en la ruta /data para enviar los datos de los movimientos, esto se puede ver en la Ilustración 5-4, donde se ven las solicitudes de tipo POST y que por cada una de esas solicitudes se generan algunos paquetes acks. Esto se debe a que HTTP trabaja sobre TCP y este último lo que está realizando es una conexión de 3 pasos llamados handshake de 3 vías, que en la captura se lo puede ver como ack y sync ack.

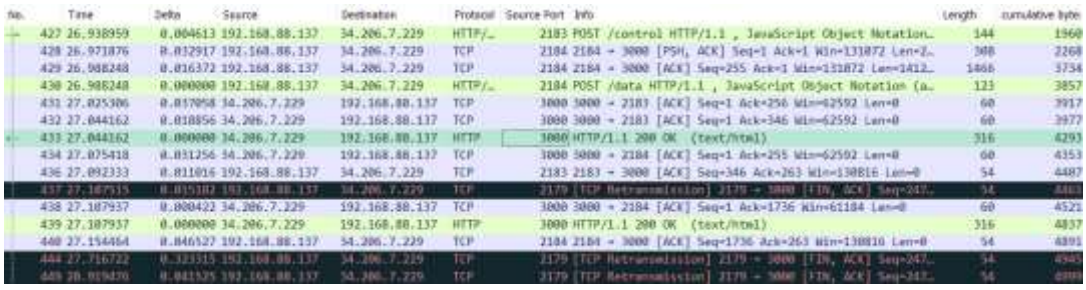


Ilustración 5-4: Captura de paquetes cuando el estudiante mueve el brazo robótico.

Realizado por: Huilcamaygua, Cinthya, 2023.

4.2 Obtención de parámetros de rendimiento a partir de una práctica de laboratorio

Para cualquiera de los dos escenarios, se coloca el filtro en Wireshark: ((ip.src == 34.206.7.229) or (ip.dst == 34.206.7.229)) and not (tcp.port == 22 or udp.port == 22 or tcp.port == 8080) para analizar el tráfico específico entre el usuario y el servidor. En la Ilustración 6-4, se puede observar que se ha filtrado todo el tráfico concerniente a los paquetes que provienen o van hacia la dirección IP del servidor.

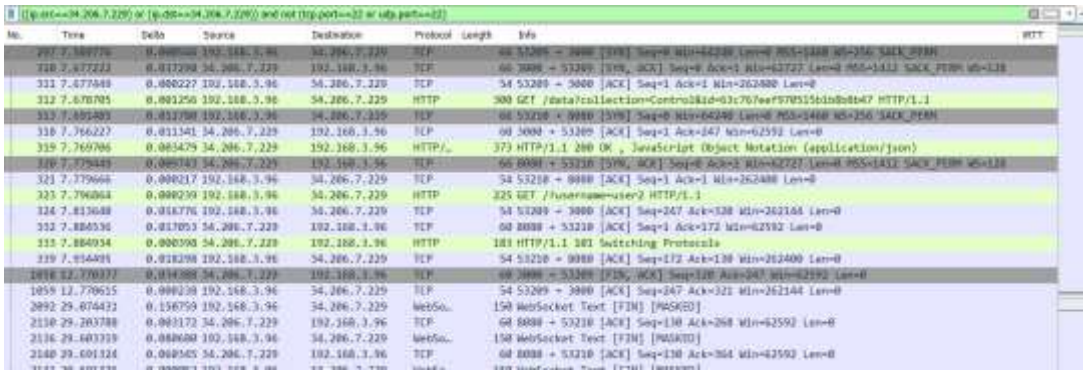


Ilustración 6-4: Captura de tráfico de los paquetes entre el usuario y servidor

Realizado por: Huilcamaygua, Cinthya, 2023.

4.2.1 Ancho de banda

El ancho de banda (bits/s) se lo puede encontrar en la opción Conversaciones de la pestaña Estadísticas, en esa ventana se selecciona la pestaña Ethernet y en las dos últimas columnas estaría el tráfico tanto de subida como de bajada. En la Ilustración 7-4, el ancho de banda de subida es de 618 bits/s y el de bajada es de 291 bits/s.

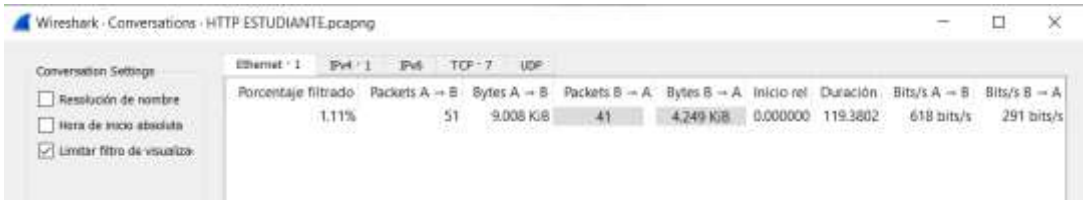


Ilustración 7-4: Obtención de ancho de banda

Realizado por: Huilcamaygua, Cinthya, 2023.

El ancho de banda total seria la suma entre el ancho de banda de subida como de bajada dando un resultado de 909 bits/s.

4.2.2 Tiempo de ida y vuelta

4.2.2.1 Para el escenario HTTP

En este escenario utilizando HTTP solamente existen las solicitudes generadas dentro del ambiente de realidad virtual como el envío de los movimientos de cada uno de los intentos que tiene el estudiante. Estas peticiones se envían cada vez que el estudiante interactúa con los botones del menú del brazo robótico.

Para medir el tiempo que lleva transmitir los datos, se accede a la función "Conversaciones" luego, a la pestaña TCP en donde se hallaría varias filas de solicitudes realizadas. En la Ilustración 9-4, cada una de estas filas generan una gráfica del tiempo de respuesta (RTT) para cada ID de flujo (Stream ID). Cada ID de flujo está asociado a una conversación originada por HTTP, ya que una característica propia de HTTP es la creación de una nueva conexión para cada solicitud efectuada.

Ethernet · 1 | IPv4 · 1 | IPv6 | TCP · 7 | UDP

Dirección A	Puerto A	Dirección B	Puerto B	uetes	Bytes	Stream ID	otales	je filtrado	Packets A → B	Bytes A → B	Paci
192.168.3.96	53209	34.206.7.229	3000	11	1.173 KiB	21	11	100.00%	6	582 bytes	
192.168.3.96	53233	34.206.7.229	3000	14	2.703 KiB	81	14	100.00%	8	2.096 KiB	
192.168.3.96	53234	34.206.7.229	3000	13	1.325 KiB	82	13	100.00%	7	735 bytes	
192.168.3.96	53239	34.206.7.229	3000	13	1.325 KiB	89	13	100.00%	7	735 bytes	
192.168.3.96	53240	34.206.7.229	3000	14	2.702 KiB	90	14	100.00%	8	2.095 KiB	
192.168.3.96	53243	34.206.7.229	3000	14	2.703 KiB	94	14	100.00%	8	2.096 KiB	
192.168.3.96	53244	34.206.7.229	3000	13	1.325 KiB	95	13	100.00%	7	735 bytes	

Ilustración 8-4: Ventana conversaciones en la pestaña TCP del tráfico filtrado para HTTP.
Realizado por: Huilcamaygua, Cinthya, 2023.

Se unificaron todas estas solicitudes en un solo gráfico para tener una idea clara de todas estas solicitudes dentro del evento con el escenario HTTP. En la Ilustración 10-4 están las gráficas tanto de las solicitudes hacia el servidor y las respuestas del servidor hacia el usuario estudiante.

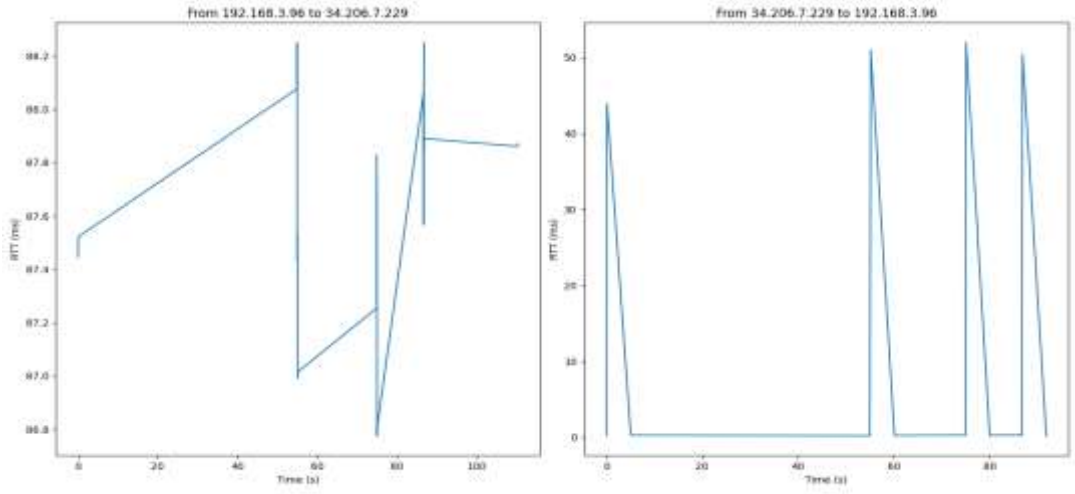


Ilustración 9-4: Gráfico del tiempo de respuesta para el tráfico generado por las solicitudes del estudiante en el escenario con HTTP.

Realizado por: Huilcamaygua, Cinthya, 2023.

En las gráficas se ven claramente las peticiones realizadas desde el usuario estudiante hacia el servidor y viceversa. En el gráfico de la izquierda cada pico representa la solicitud del estudiante al servidor cuando inicia la aplicación de realidad virtual y cuando postea los movimientos del brazo robótico. Se puede ver que el tiempo de respuesta ronda los 80ms y 90ms. Mientras que, en el gráfico del lado derecho están las respuestas a las solicitudes del estudiante, que en este caso vendría a ser la conexión al servidor y los otros 3 picos serían los envíos del estado del número de intentos, estas peticiones tendrían un retardo menor a lo 52ms.

Cada pico en el grafico derecho es la respuesta de un POST que ha realizado el estudiante, mientras que en el grafico izquierdo el estudiante la mayoría de las peticiones son de tipo GET.

En el caso de uso de la aplicación del profesor, se han realizado distintas peticiones, en la Ilustración 11-4, en el grafico izquierdo cada punto o pico de la gráfica es una petición HTTP que corresponde a una solicitud GET, estas peticiones GET no se verían reflejados en las respuestas del servidor, pues en el grafico del lado derecho se puede observar que existe un pico y este corresponde a una solicitud tipo POST en donde el servidor recibió el dato del número de intentos para ser almacenado en la base de datos

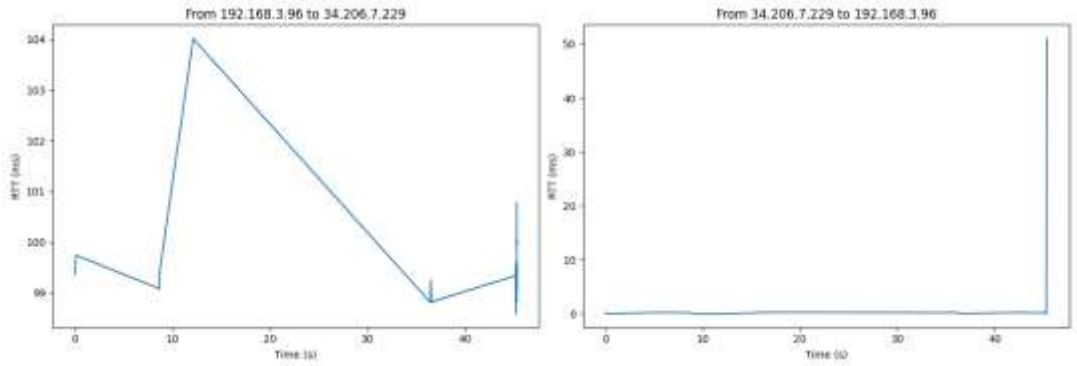


Ilustración 10-4: Gráfico del tiempo de ida y vuelta para el tráfico generado en el escenario con HTTP.

Realizado por: Huilcamaygua, Cinthya, 2023.

4.2.2.2 Para MQTT

En el escenario con MQTT fue más sencillo obtener el tiempo de ida y vuelta, dado que en Wireshark se muestra una conversación por cada tópico generado en la aplicación de realidad virtual y su gráfica se visualiza tal como se muestra en la Ilustración 11-4.

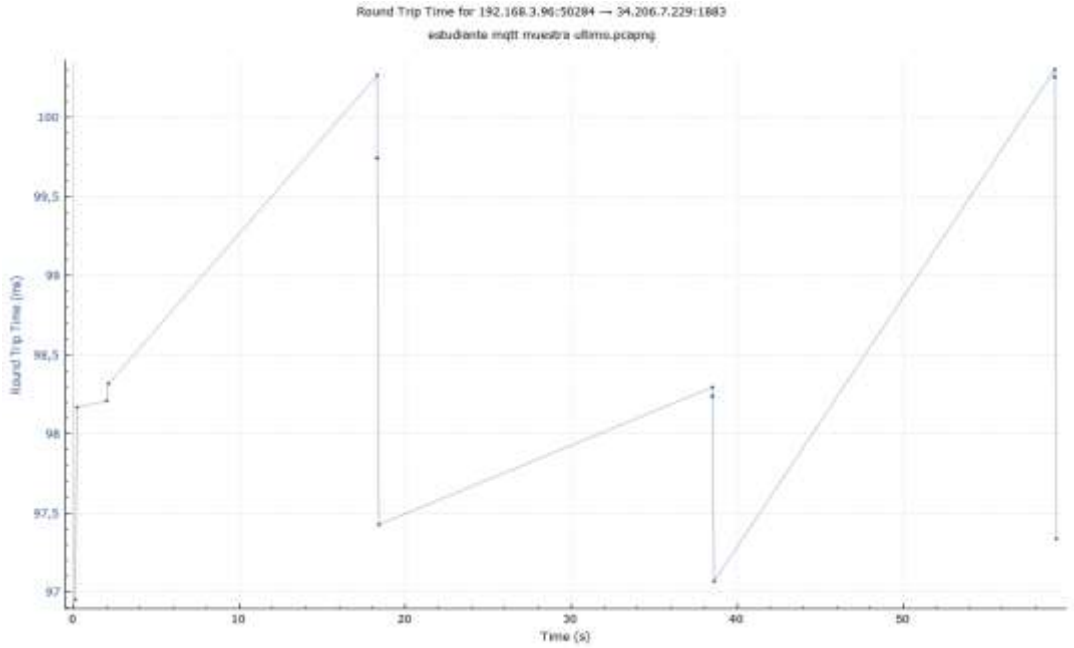


Ilustración 11-4: Gráfico del tiempo de ida y vuelta para el tráfico generado en el escenario MQTT.

Realizado por: Huilcamaygua, Cinthya, 2023.

Se toman todos los valores que se muestran en la gráfica en una tabla para obtener los valores mínimos, los máximos, y el valor promedio de toda la práctica de laboratorio.

4.2.3 Pérdida de paquetes

Dado que, tanto HTTP como MQTT se comunican bajo el protocolo TCP, es posible hallar las pérdidas de paquetes como errores de transmisión en la gráfica de E/S de Wireshark.

En la Ilustración 12-4, se ha realizado la captura de tráfico del escenario de HTTP y se puede observar que no existen errores TCP en toda la comunicación entre el usuario estudiante y el servidor. De manera que el porcentaje de error o pérdidas de paquetes que nos indica la gráfica es de 0%.

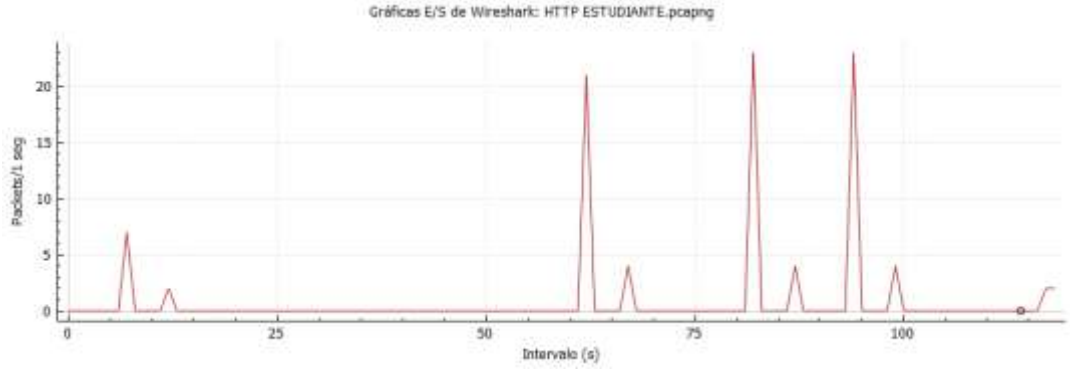


Ilustración 12-4: Gráfico de E/S de Wireshark con el filtro para errores TCP.

Realizado por: Huilcamaygua, Cinthya, 2023.

Mientras que, en la aplicación del profesor, si hubo algunos errores de transmisión, pero no existieron pérdidas de paquetes, los paquetes que se han perdido son de 7 paquetes según la Ilustración 13-4 en todo el tiempo que duró el evento. Estos errores de paquetes corresponderían a retransmisiones de TCP.

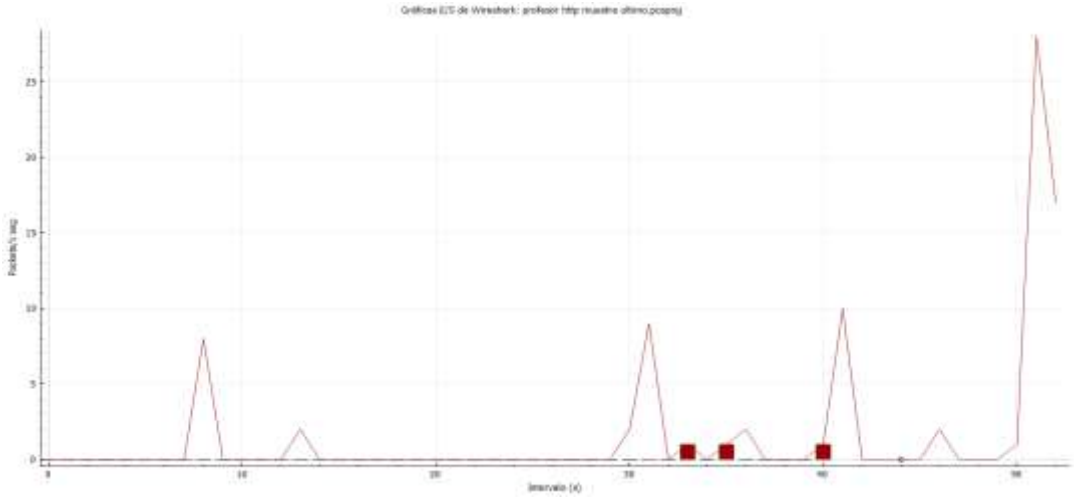


Ilustración 13-4: Gráfico E/S de Wireshark para el tráfico generado por las solicitudes de HTTP del profesor.

Realizado por: Huilcamaygua, Cinthya, 2023.

Es así como, el porcentaje de errores de TCP sería el valor de 3 por sobre los 96 paquetes HTTP filtrados. Dando un valor de 3.12% de los paquetes con errores. Este error puede variar dado que se ha tomado de una prueba piloto.

4.2.4 Uso de recursos CPU y RAM del servidor

Se ha tomado en cuenta el uso de CPU y RAM del servidor pues son parámetros importantes si se quiere saber el rendimiento de una red. El servidor es el que gestiona toda información generada por los estudiantes y también las peticiones de las funciones de las aplicaciones de profesor y

estudiantes por lo que medir estos recursos es importante. Las gráficas se obtendrían a través del servicio de CloudWatch de AWS.

4.2.4.1 En el escenario usando MQTT

Las gráficas generadas por CloudWatch permite la visualización del uso de los recursos RAM y CPU a lo largo del tiempo por lo que es fácil visualizar los cambios. El escenario con el protocolo MQTT se puso en marcha desde las 23:16 horas por lo que se ha buscado la hora exacta en dichas gráficas. (Ver Ilustración 14-4)

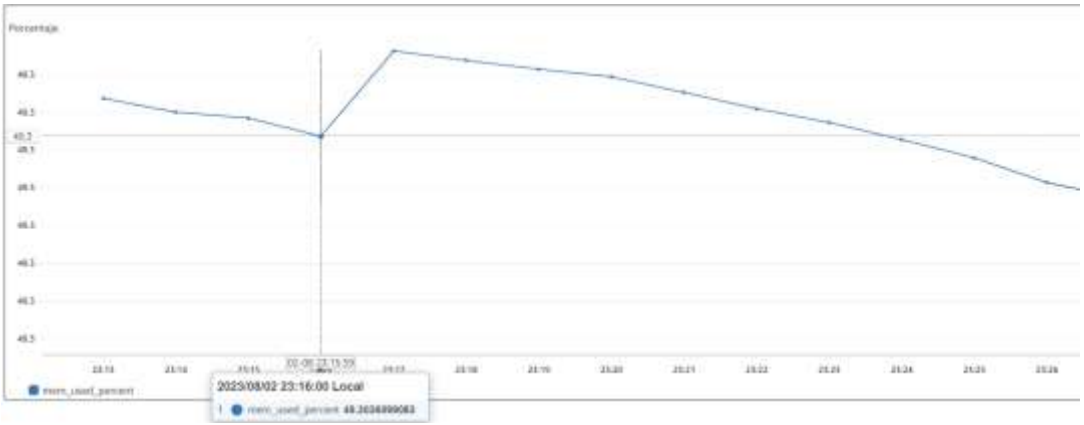


Ilustración 14-4: Porcentaje de uso de RAM del servidor AWS para el escenario MQTT

Realizado por: Huilcamaygua, Cinthya, 2023.

En la Ilustración 15-4, se puede observar que en el escenario de MQTT el consumo de memoria RAM fue de 49.3% y llego a un pico de 49.34%, la variación de este consumo es mínimo con una diferencia de 0.04% en el tiempo que sucedió el evento.

Mientras que el uso del CPU También fue mínimo, tomando en cuenta que el evento inicio a las 23:16 horas, el uso de la CPU presenta una leve variación de 1.9% hasta los 1.93% con una diferencia de 0.03%, tal como se puede observar en la Ilustración 15-4.

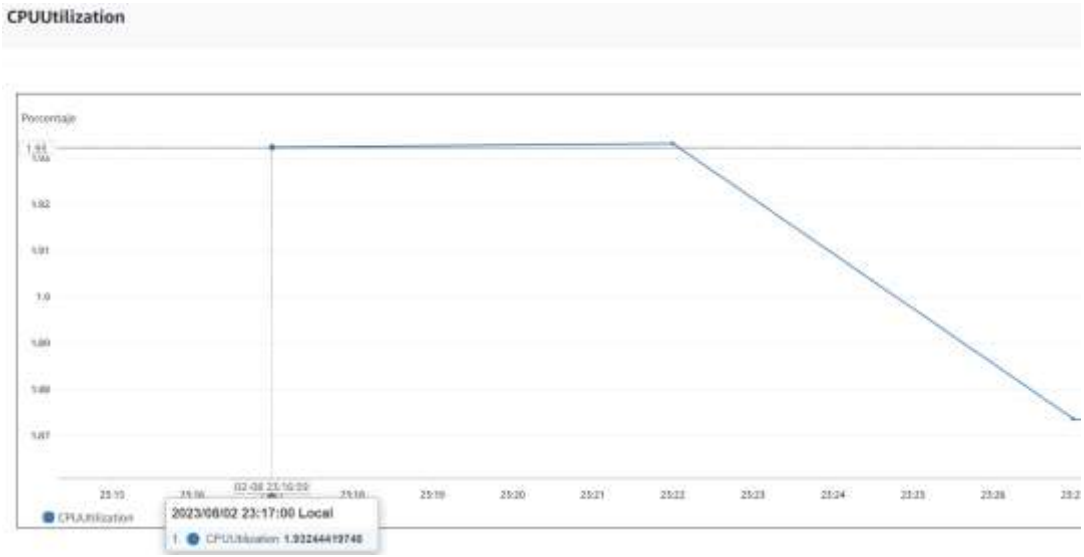


Ilustración 15-4: Porcentaje de uso de CPU del servidor AWS para el escenario MQTT

Realizado por: Huilcamaygua, Cinthya, 2023.

4.2.4.2 En el escenario usando HTTP

Se utilizó el servicio de CloudWatch para visualizar el uso de RAM y CPU tal como para el escenario con MQTT. En la ilustración 16-4, se puede observar que la hora de inicio fue a las 13:05 horas con un porcentaje de uso de 45.9% hasta llegar a un pico de 52.2% con una diferencia de 5.9 puntos porcentuales.

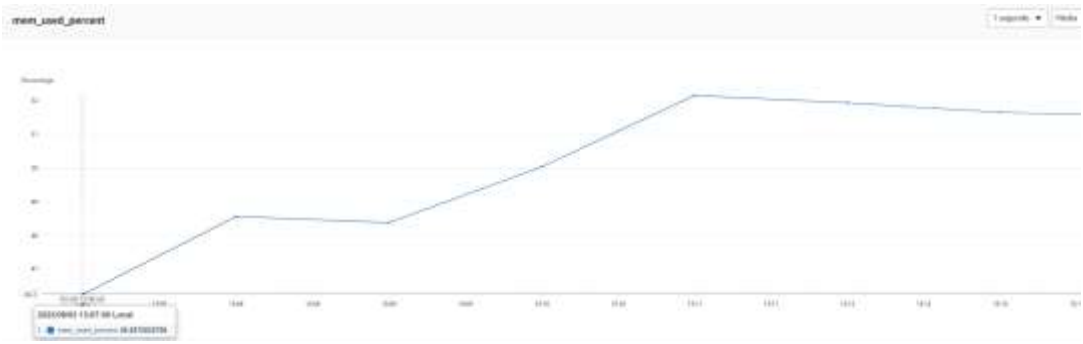


Ilustración 16-4: Porcentaje de uso de RAM del servidor AWS para el escenario HTTP.

Realizado por: Huilcamaygua, Cinthya, 2023.

Ahora, la gráfica de porcentaje de uso de CPU también presenta variaciones, el valor empieza en los 2.5% y llega a un pico de 2.96% con una diferencia entre pico y mínimo de 0.46%. Como se puede observar en la Ilustración 17-4.

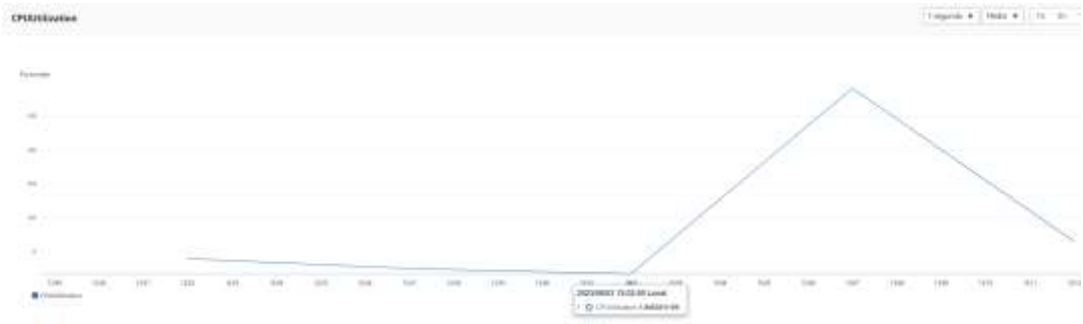


Ilustración 17-4: Porcentaje de uso de CPU del servidor AWS para el escenario HTTP.

Realizado por: Huilcamaygua, Cinthya, 2023.

4.3 Observaciones a lo largo de 24 horas

Una vez obtenidos los datos de Wireshark, se colocan a todos dentro de gráficas y ver el comportamiento durante 1 día entero. Son 30 las muestras que se tomarían a lo largo de un día en lapsos de 48 minutos en un día.

4.3.1 Escenario con MQTT

En las pruebas realizadas a lo largo de un día, se pudo obtener el caso cuando la conexión a internet se encuentra descongestionada y cuando el sistema se encuentra más congestionado presentando un valor de RTT más elevado, esto se lo puede ver en la Ilustración 18-4.

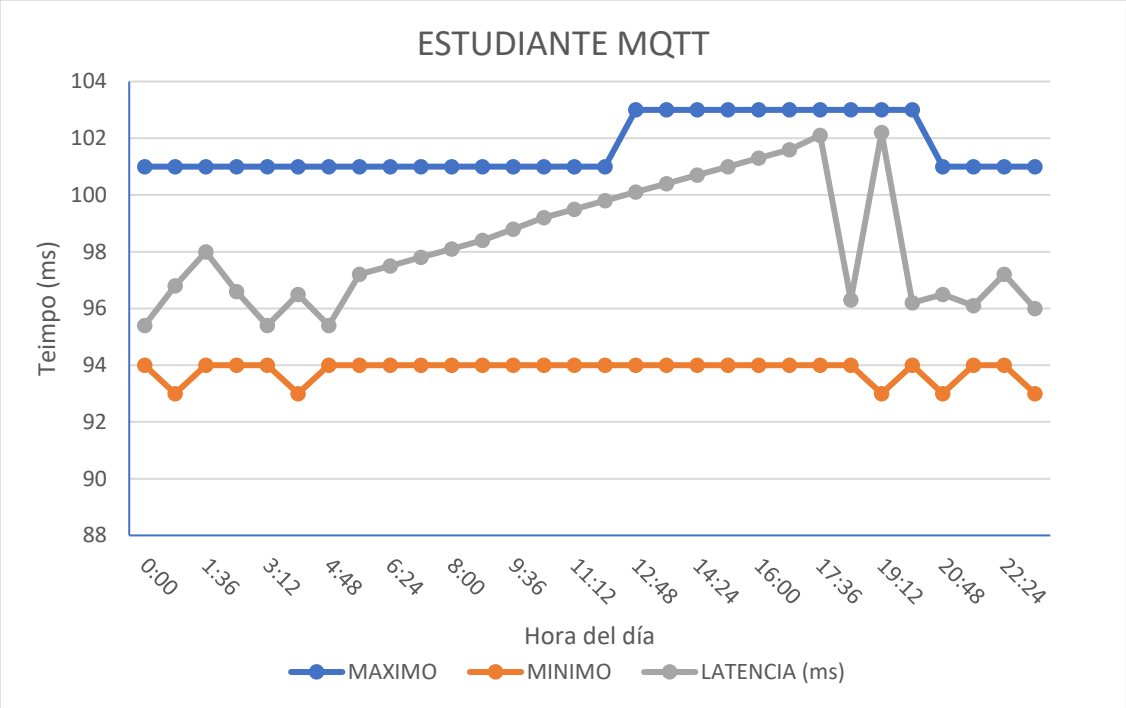


Ilustración 18-4: Gráfica de RTT máxima, mínima y promedio de las mediciones realizadas a lo largo de 24 horas

Realizado por: Huilcamaygua, Cinthya, 2023.

También se realizó este mismo análisis con el escenario del protocolo HTTP ejecutando los 3 intentos que tiene la práctica de laboratorio.

Así mismo, en la Ilustración 19-4 se encuentra el valor de RTT promedio y los valores máximos y mínimos que existieron en las 30 mediciones.

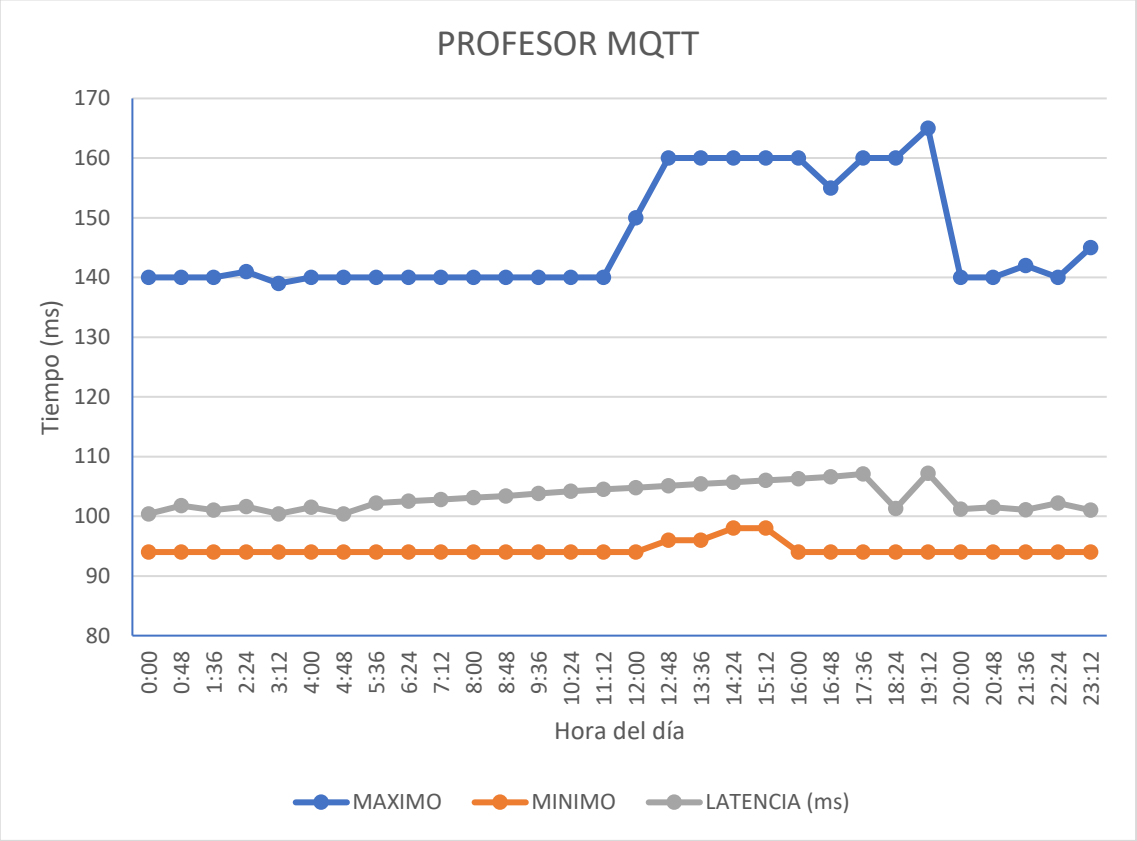


Ilustración 19-4: Gráfica de la latencia máxima, mínima y promedio de las mediciones realizadas a lo largo de 24 horas en el escenario HTTP.

Realizado por: Huilcamaygua, Cinthya, 2023.

El valor de la latencia promedio de cada medición se puede ver que es casi siempre la misma a lo largo del día, tiene un valor entre los 100ma y los 107,2 ms.
Las Tablas usadas se encuentra en el Anexo F donde se encuentran todos los datos para obtener las gráficas.

4.3.2 Escenario con HTTP

La grafica obtenida a lo largo del día para la aplicación del estudiante en valores de latencia indican valores promedio que forman una gráfica lineal sin muchas variaciones. En la Ilustración 20-4, se puede ver que los promedios de cada medición esta entre los 98ms y los 104,5ms.

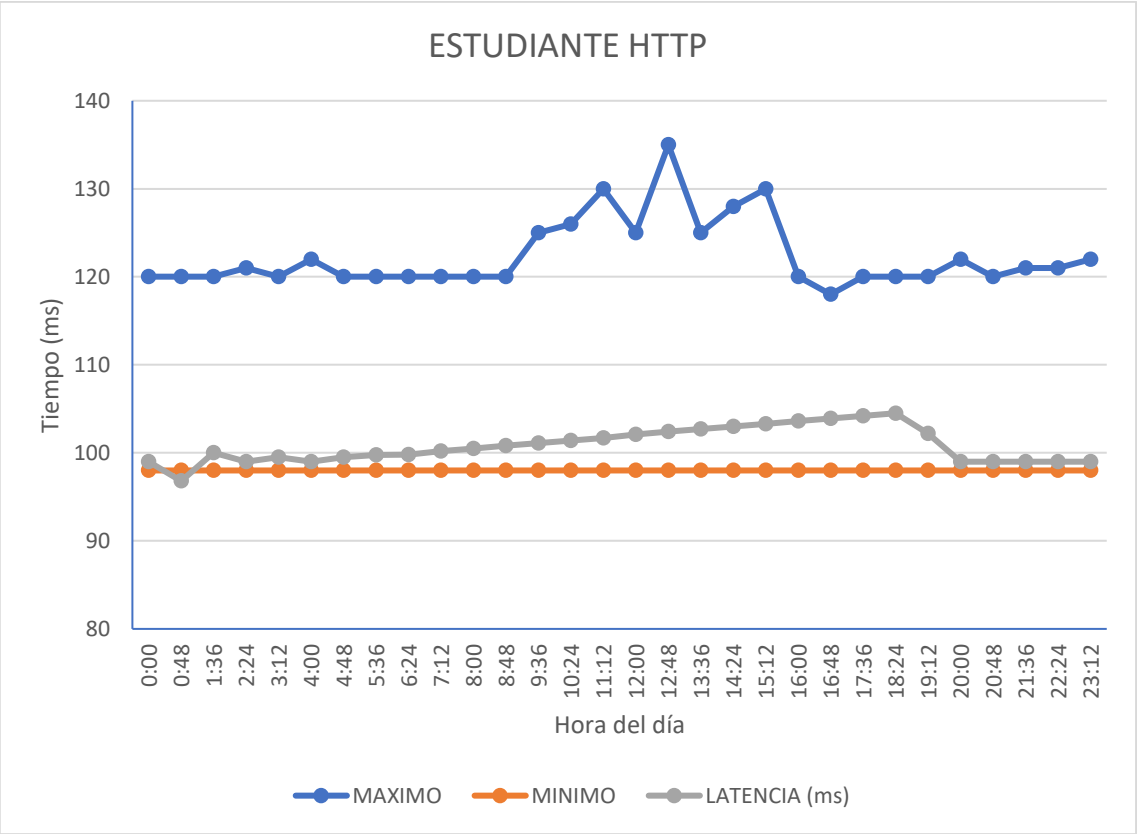


Ilustración 20-4: Gráfica de la latencia máxima, mínima y promedio de las mediciones realizadas a lo largo de 24 horas en el escenario HTTP.

Realizado por: Huilcamaygua, Cinthya, 2023.

En la Ilustración 21-4, se puede observar que, para el caso del profesor, cuando realiza las solicitudes GET en las mediciones realizadas a lo largo del día, la tendencia aumenta en las horas pico del día, pero casi siempre tiene un valor de 109ms.

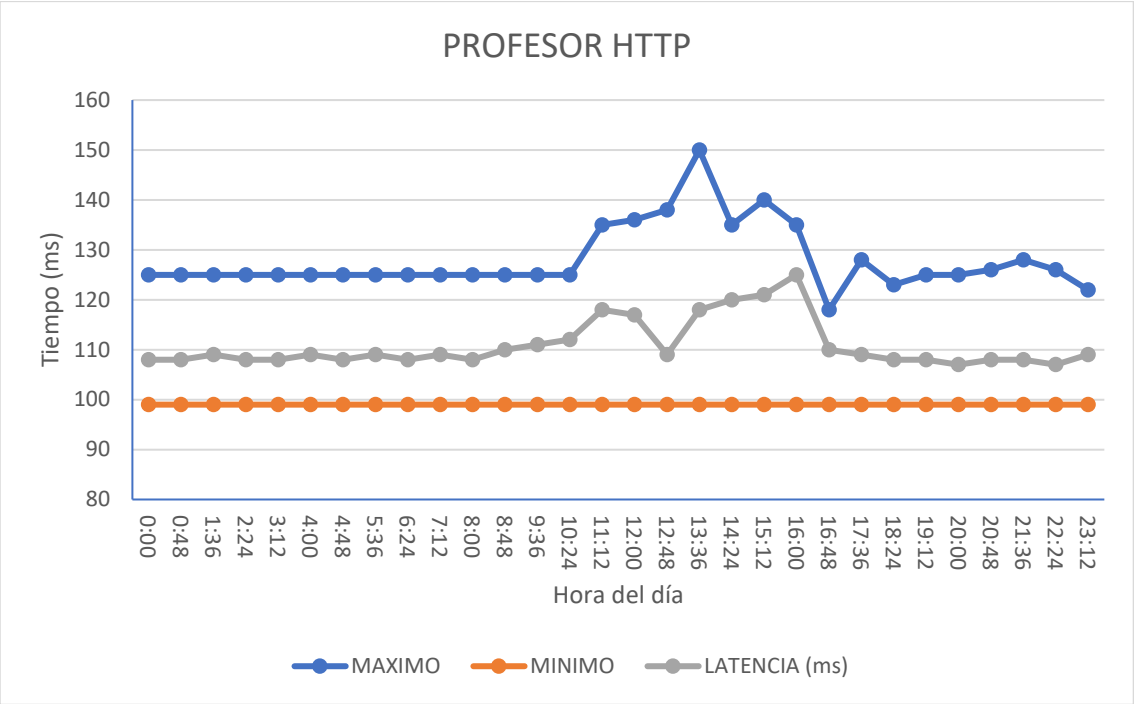


Ilustración 21-4: Gráfica de la latencia máxima, mínima y promedio de las mediciones realizadas a lo largo de 24 horas en el escenario HTTP.

Realizado por: Huilcamaygua, Cinthya, 2023.

4.4 Discusión de resultados

Aunque se recopilaron datos sobre parámetros como RTT, pérdida de paquetes y ancho de banda utilizado para cada usuario en cada escenario, individualmente, no proporcionan una indicación clara de la eficiencia del escenario. La determinación de la eficiencia del escenario se logra al comparar los parámetros de rendimiento obtenidos en cada uno de los escenarios propuestos. Este análisis permite evaluar y determinar cuál de los dos escenarios tuvo un rendimiento superior.
Para esto se han realizado 30 observaciones, cada observación corresponde a una práctica de laboratorio donde no se ejecutan interacciones en tiempo real, solo funciones de almacenado de datos.

En este contexto, es importante señalar que tanto el estudiante como el profesor llevan a cabo actividades específicas. El estudiante interactúa con el brazo robótico y guarda las posiciones hasta completar los tres intentos, mientras que el profesor revisa los datos o movimientos almacenados por el estudiante y restaura o reinicia el número de intentos. Además, en ambos escenarios de red, realizan las mismas actividades con el propósito de mantener un entorno controlado bajo condiciones idénticas:

- Para el jugador estudiante:

Dentro de los primeros 10 segundos, se captura la conexión hacia el servidor en la aplicación de realidad virtual. A los 20, 40 y 60 segundos, se envían tres movimientos del brazo robótico correspondientes a cada intento en ambos escenarios.

- Para el jugador profesor:

Dentro de los primeros 10 segundos, se captura la conexión hacia el servidor en la aplicación de realidad virtual. A los 20, 40 y 60 segundos, se revisa cada intento del brazo robótico.

4.4.1 Comparación de resultados

La comparación de resultados entre los escenarios comprende solamente las gráficas de los valores promedios de las observaciones a lo largo de las 24 horas.

En la Ilustración 22-4, se observa el gráfico de ambos escenarios en función del tiempo de ida y vuelta para el usuario estudiante, en donde la hora menos congestionada es desde las 00:00 hasta las 05:36 horas del día. Y el horario más crítico se halla entre las horas del medio desde las 10:24 hasta las 20:00 horas.

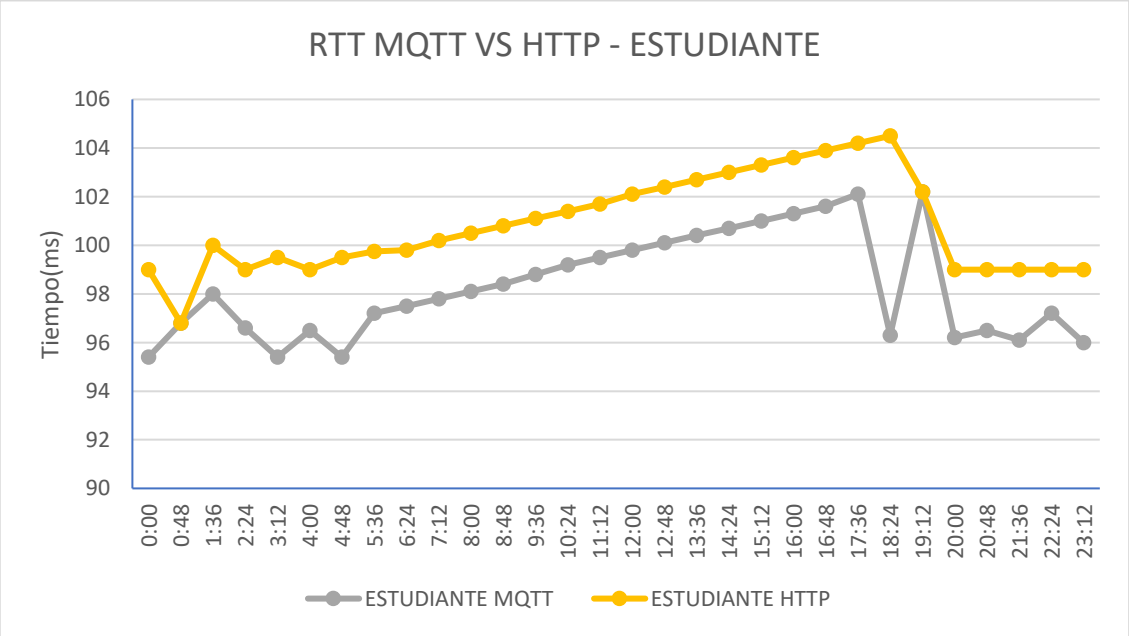


Ilustración 22-4: Gráfica de RTT de las 28 mediciones en los escenarios MQTT y HTTP.

Realizado por: Huilcamaygua, Cinthya, 2023.

También se puede observar que el RTT en el escenario de HTTP es mayor al obtenido con MQTT bajo las mismas condiciones.

En la Ilustración 23-4, se observa el gráfico de ambos escenarios en función del tiempo de ida y vuelta para el usuario profesor, en donde la hora menos congestionada es desde las 00:00 hasta las 05:36 horas del día. Y el horario más crítico se halla entre las horas del medio desde las 10:24 hasta las 20:00 horas.

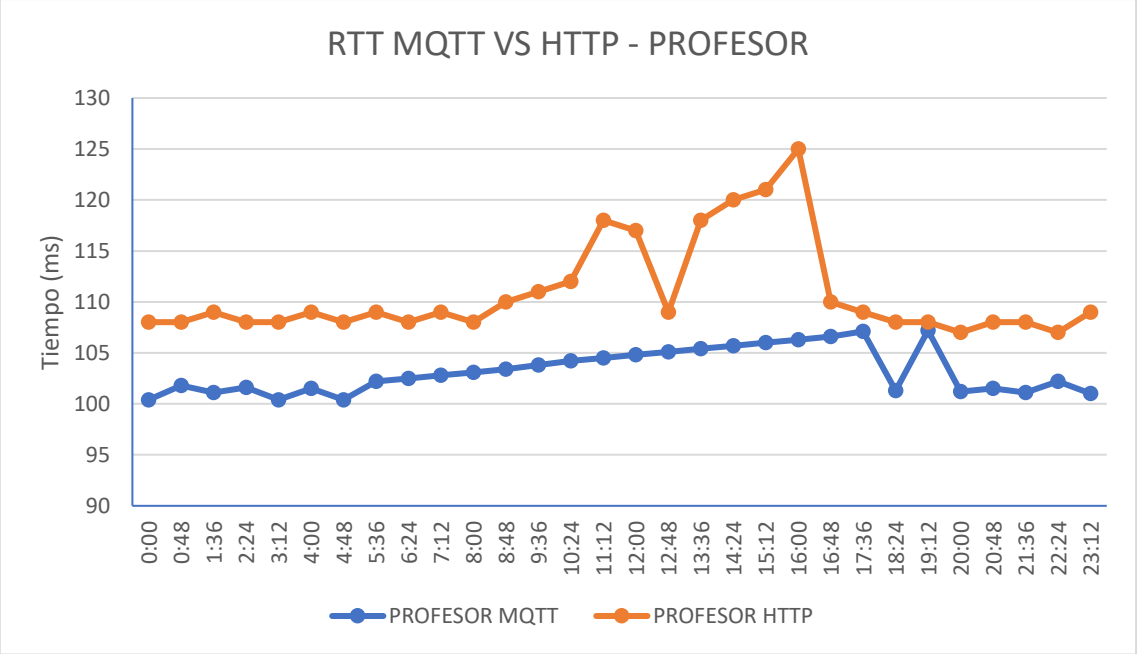


Ilustración 23-4: Gráfica de RTT de las 28 mediciones en los escenarios MQTT y HTTP.

Realizado por: Huilcamaygua, Cinthya, 2023.

En cuanto a los errores del transmisión, para esos mismos usuarios, se presentan en la Ilustración 24-4, donde se encuentran los datos del porcentaje de errores para los usuarios estudiante y profesor de cada escenario en una sola gráfica. Dado que al enviarse sobre TCP no se van a perder paquetes, pero si habrá un porcentaje de retransmisiones. Se observa claramente que los usuarios con MQTT no presentan errores de retransmisión, mientras que con el protocolo HTTP los errores de transmisión van desde el 2% hasta el 3% de errores durante toda la comunicación con el servidor.

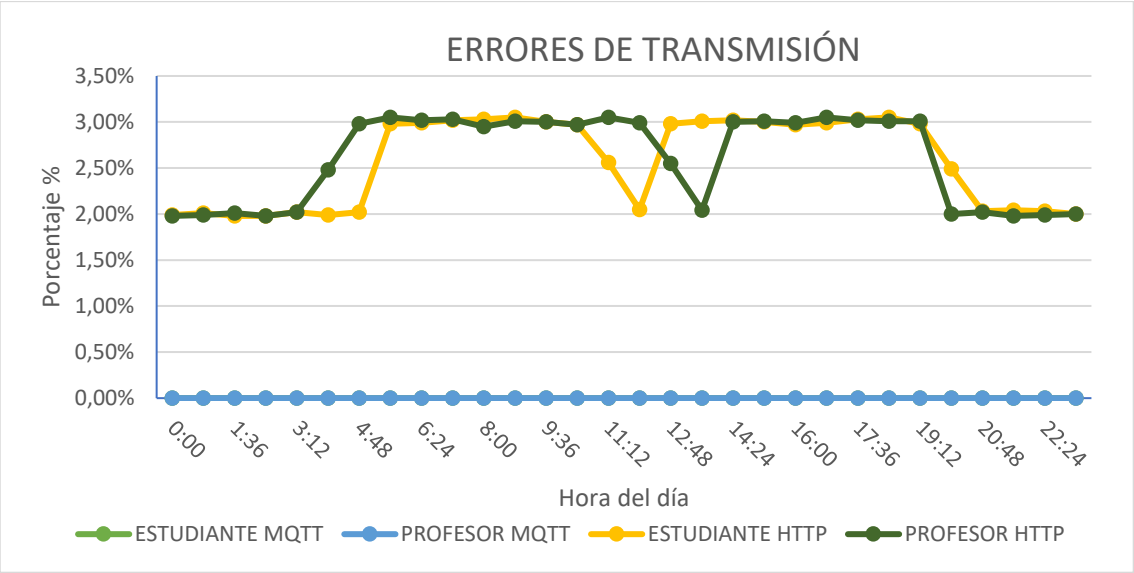


Ilustración 24-4: Gráfica errores en la comunicación promedio de las mediciones realizadas a lo largo de 24 horas en el escenario HTTP y MQTT.

Realizado por: Huilcamaygua, Cinthya, 2023.

Mientras tanto, el ancho de banda utilizado por los dos usuarios en ambos ambientes si varían en las horas pico. En la Ilustración 25-4, se puede ver que tanto el estudiante como el profesor en HTTP utilizan mayor ancho de banda que los usuarios profesor y estudiante en MQTT. Además, al ser una escala muy grande la gráfica pudiera verse como una línea recta, aunque los datos si demuestren variación en diferentes horarios en los que se ha tomado la muestra

En la Ilustración 25-4, se detalla el ancho de banda consumido durante la práctica de laboratorio.

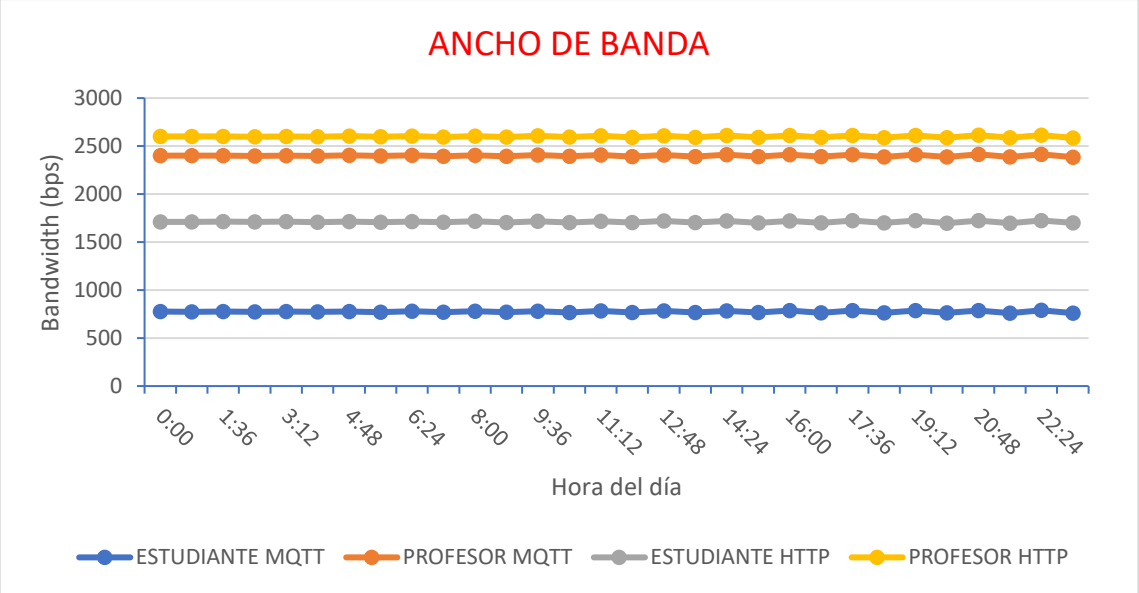


Ilustración 25-4: Gráfica de ancho de banda a lo largo de 24 horas tanto en el escenario HTTP como MQTT.

Realizado por: Huilcamaygua, Cinthya, 2023.

4.4.2 Discusión

En la Tabla 1-4 se puede ver las diferencias porcentuales de latencia error y ancho de banda adicionalmente el porcentaje de red que necesita la aplicación de acuerdo con el ancho de banda del proveedor de internet antes mencionado de 20 Mbps

Tabla 1-4: Tabla comparativa de los parámetros de rendimiento.

MQTT VS HTTP - APLICACIÓN ESTUDIANTE							
RTT		% Error		Ancho de banda		Uso de la red de 20Mbps	
RTT Promedio MQTT (ms)	98.27	MQTT	0%	Ancho de Banda Promedio MQTT (bps)	774.50	Con MQTT	0.003873%
RTT Promedio HTTP (ms)	100.83	HTTP	2%	Ancho de Banda Promedio HTTP (bps)	1710.63	Con HTTP	0.008550%
Diferencia (ms)	2.56			Diferencia (bps)	936.13		
Diferencia Porcentual de Latencia	2.61%			Diferencia Porcentual	120.9%		

Realizado por: Huilcamaygua, Cinthya, 2023.

Tabla 2-4: Características del software de realidad virtual.

MQTT VS HTTP - APLICACIÓN PROFESOR							
RTT		% Error		Ancho de banda		Uso de la red de 20Mbps	
RTT Promedio MQTT (ms)	103.40	MQTT	0%	Ancho de Banda Promedio MQTT (bps)	2399.5	Con MQTT	0.012000%
RTT Promedio HTTP (ms)	110.90	HTTP	3%	Ancho de Banda Promedio HTTP (bps)	3399	Con HTTP	0.013000%
Diferencia (ms)	7.50			Diferencia (bps)	999.5		
Diferencia Porcentual de Latencia	7.00%			Diferencia Porcentual	42%		

Realizado por: Huilcamaygua, Cinthya, 2023.

Tanto en la aplicación del estudiante como al del profesor se evidencia un aumento significativo en uso de ancho de banda y el porcentaje de error siendo MQTT un valor cercano a 0% y HTTP a un 3%

También se tiene en cuenta el uso de los recursos CPU y RAM del servidor. Los valores promedios que se obtuvieron no cambiaron a lo largo del tiempo.

Lo que sí se puede destacar es que la variación de RAM promedio entre MQTT y HTTP es que con HTTP el servidor consume 5.8% más que con MQTT; es decir, que si MQTT consume un 45.6% de recursos RAM, HTTP consume 51.4%. Mientras que el recurso CPU se mantuvo estable

en el valor 2.85% de su uso tanto en MQTT como en HTTP para el usuario estudiante. Mientras que para el profesor, la diferencia entre el protocolo MQTT y HTTP en términos de RAM es de 2.3% y en CPU consumen casi lo mismo con un porcentaje de 3.3% a 3.4%

CAPÍTULO V

5. MARCO PROPOSITIVO

5.1 PROPUESTA

En etapas previas de la investigación, identificamos que el protocolo HTTP tradicional presenta ciertas limitaciones para aplicaciones que requieren una comunicación en tiempo real. Específicamente, HTTP carece de la capacidad intrínseca para mantener un canal abierto a flujos de datos, es decir carece de la capacidad de contestar sin ninguna solicitud desde el usuario final es esencial para simular y establecer una comunicación en tiempo real.

WebSocket es un protocolo que permite interacciones bidireccionales en tiempo real entre el servidor y el cliente, superando las limitaciones de HTTP en cuanto a la función de suscripción. Adicionalmente, WebSocket se integra bien con las tecnologías web actuales, lo que facilitaría su implementación en una variedad de plataformas y dispositivos. Su capacidad para mantener una conexión abierta y permitir la transmisión de datos sin la necesidad de reestablecer conexiones recurrentemente lo hace particularmente atractivo para nuestro propósito y realizar mayores pruebas con este protocolo.

La estructura utilizada para la comunicación sigue siendo la misma de HTTP (Ilustración 1-5), es una red centrada en servidor y el software que gestione los mensajes sería node.js.

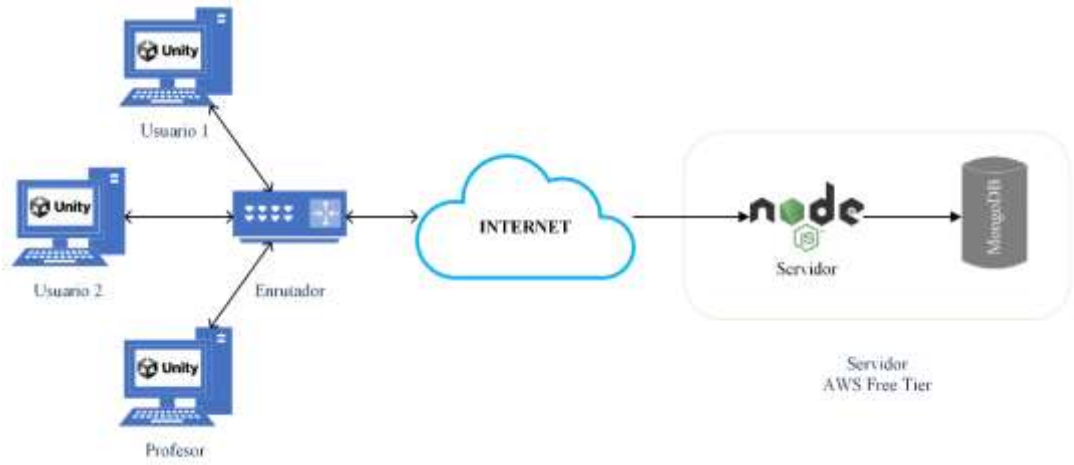


Ilustración 1-5: Topología de red centrada en servidor

Realizado por: Huilcamaygua, Cinthya, 2023.

La aplicación realizada en node.js es similar a la de HTTP pues puede escuchar en el puerto 3000 para realizar la conexión y luego se conecta al puerto 2432 para realizar la comunicación con WebSocket.

De manera que se realizó la toma de datos con el mismo número de muestras a lo largo de 24 horas. La dinámica de recogida de datos se basa en una práctica de laboratorio que consiste en realizar diferentes acciones dentro del juego 3D en un tiempo determinado. Y para realizar una evaluación comparativa, se han ejecutado los mismos movimientos en los escenarios de MQTT y de WebSocket.

En la práctica de laboratorio se realizaron varios movimientos en los que se interactúan en tiempo real en un lapso de 15 segundos; en los primeros 5 segundos se ejecutan movimientos con un slider, en los siguientes 5 segundos se mueve el segundo slider y los últimos 5 segundos se mueve el 3er slider. Esta interacción se ejecutaría cuando se encuentren activos el estudiante y el profesor simultáneamente. Los datos recogidos se encuentran en el ANEXO G.

Los resultados se muestran en formato de gráficos, al igual que con el escenario WebSocket vs MQTT. En las Ilustración 2-5 y la Ilustración 3-5 se pueden observar los tiempos de ida y vuelta de los usuarios profesor y el estudiante.

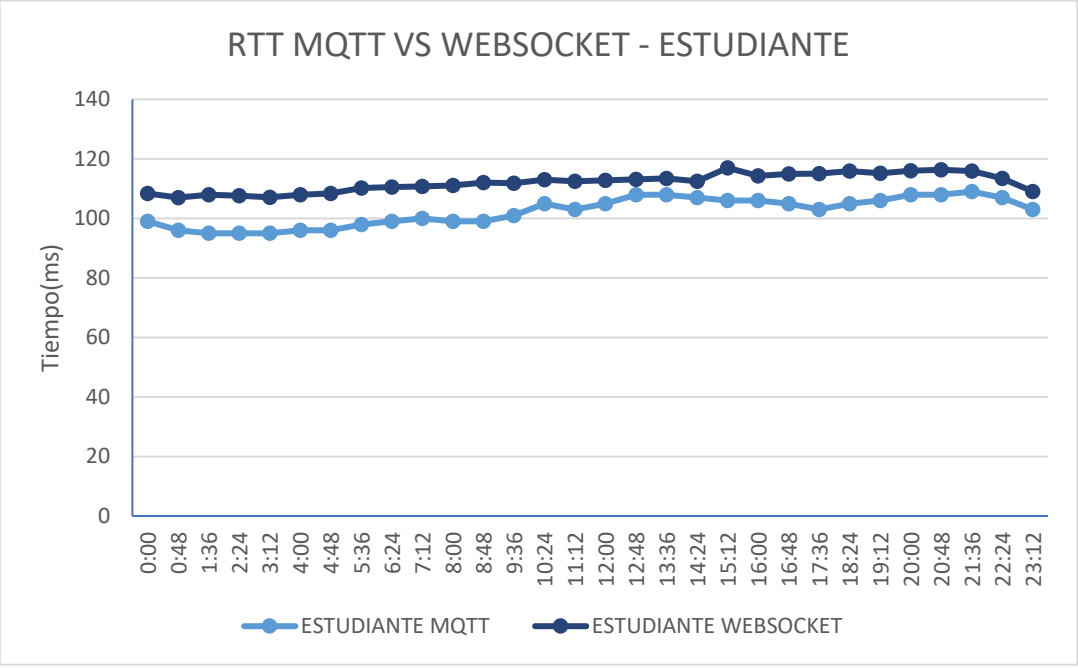


Ilustración 2-5: Gráfica de RTT MQTT vs WebSocket en la aplicación del estudiante.

Realizado por: Huilcamaygua, Cinthya, 2023.

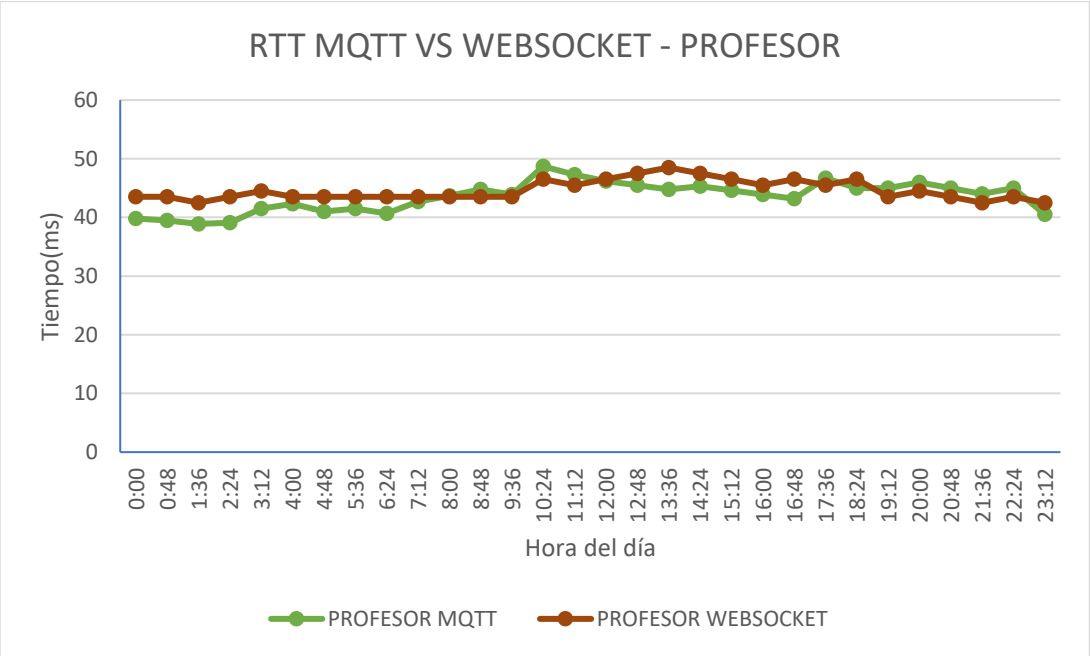


Ilustración 3-5: Gráfica de RTT MQTT vs WebSocket en la aplicación del profesor

Realizado por: Huilcamaygua, Cinthya, 2023.

En la Ilustración 4-5, se puede observar el ancho de banda consumido en cada una de las prácticas de laboratorio escenarios MQTT y WebSocket tanto del profesor como del estudiante.

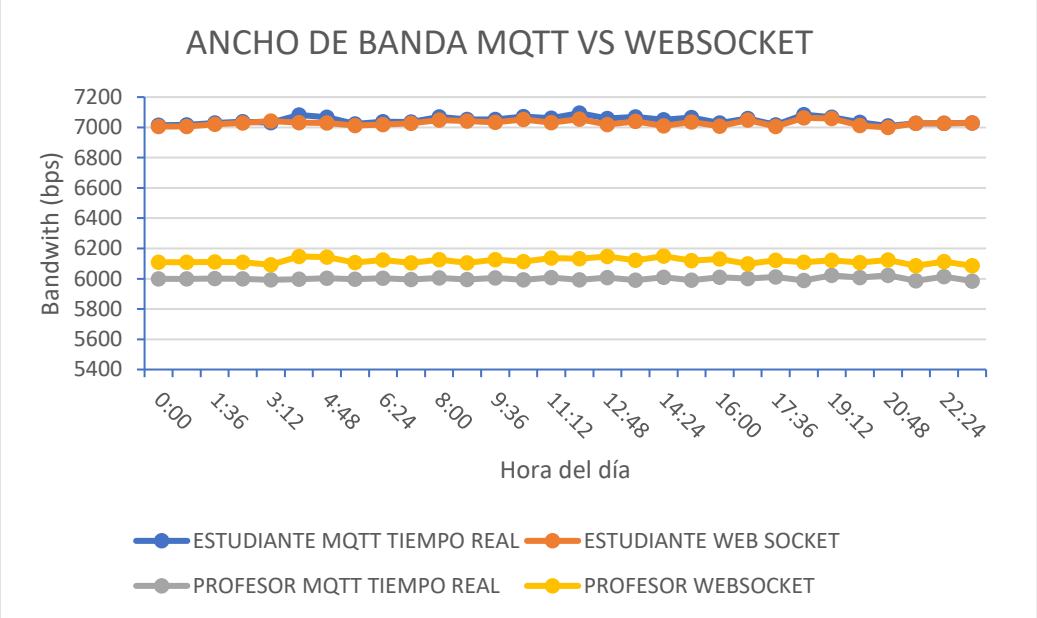


Ilustración 4-5: Ancho de banda consumido en cada practica de laboratorio para los escenarios WebSocket y MQTT.

Realizado por: Huilcamaygua, Cinthya, 2023.

Y en cuanto a los errores de transmisión, en la Ilustración 5-5 se puede ver el porcentaje de errores de transmisión que es menor al 1%.

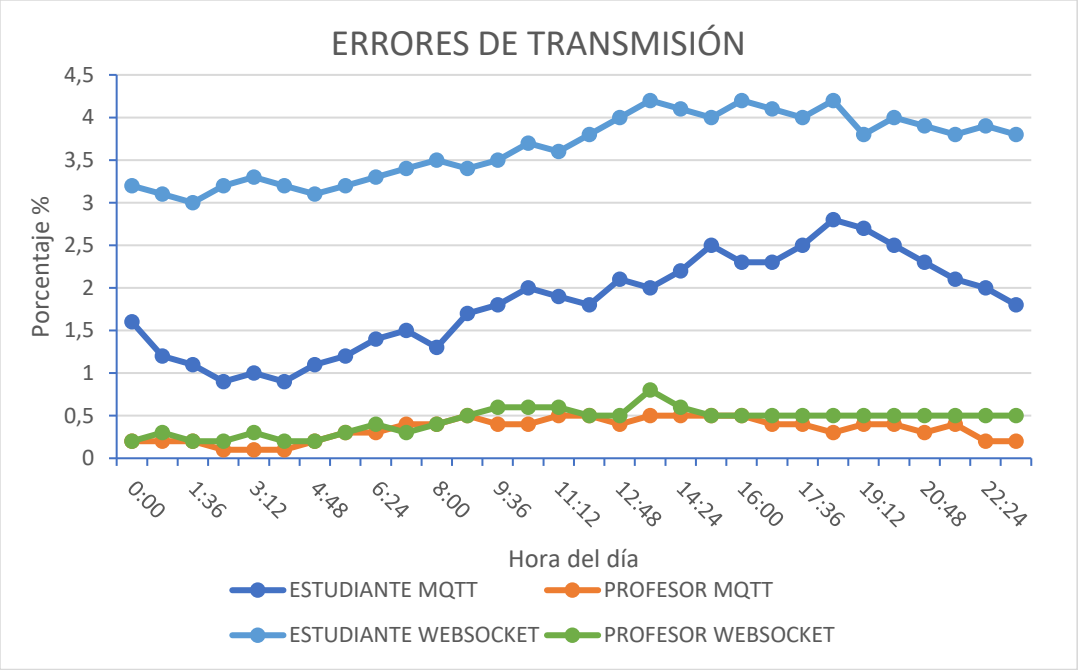


Ilustración 5-5: Errores de transmisión entre WebSocket y MQTT.

Realizado por: Huilcamaygua, Cinthya, 2023.

En la Tabla 1-5 se encuentran los datos promedios obtenidos mediante Wireshark durante la práctica de laboratorio. Se puede ver claramente que existe diferencia en cuanto al uso de MQTT o de WebSocket para realizar las mismas acciones.

Tabla 1-5: Tabla resumen de la aplicación del estudiante con WebSocket.

MQTT VS WEBSOCKET - APLICACIÓN ESTUDIANTE							
RTT		% Error		Ancho de banda		Uso de la red de 20Mbps	
RTT Promedio MQTT (ms)	102.33	MQTT	1.82%	Ancho de Banda Promedio MQTT (bps)	7028.63	Con MQTT	0.035%
RTT Promedio WebSocket (ms)	112.04	WebSocket	3.65%	Ancho de Banda Promedio WebSocket (bps)	5999.50	Con WebSocket	0.030%
Diferencia (ms)	9.71			Diferencia (bps)	1029.13		
Diferencia Porcentual de Latencia	9.48%			Diferencia Porcentual	14.3%		

Realizado por: Huilcamaygua, Cinthya, 2023.

Mientras que en la tabla 2-5, se puede ver la diferencia en cuanto al usuario servidor en ambos escenarios. El porcentaje de error es mayor en WebSocket que con MQTT. Aunque lo que más llama la atención son valores de retardo similares, con una diferencia de 7.5 ms entre MQTT y WebSocket. El ancho de banda utilizado para ambos ambientes sigue siendo WebSocket el que consume más ancho de banda en 1Kbps a diferencia de MQTT.

Tabla 2-5: Tabla resumen de la aplicación del profesor con WebSocket.

MQTT VS WEBSOCKET - APLICACIÓN PROFESOR							
RTT		% Error		Ancho de banda		Uso de la red de 20Mbps	
RTT Promedio MQTT (ms)	43.54	MQTT	0.34	Ancho de Banda Promedio MQTT (bps)	6000.9	Con MQTT	0.030%
RTT Promedio WebSocket (ms)	44.60	WebSocket	0.44	Ancho de Banda Promedio WebSocket (bps)	6099.5	Con WebSocket	0.030%
Diferencia (ms)	1.06			Diferencia (bps)	98.6		
Diferencia Porcentual de Latencia	2.44%			Diferencia Porcentual	2%		

Realizado por: Huilcamaygua, Cinthya, 2023.

En cuanto al uso de CPU y RAM, en ambos escenarios se evidenció que los valores de utilización son muy similares a lo largo de 24 horas. La utilización de CPU promedio fue de 56.22% mientras que la utilización de RAM fue de 4.979% en el ambiente MQTT. Y en WebSocket la utilización de CPU fue de 58.22% y de RAM 5.1%. De manera que en ambos escenarios la utilización de estos recursos fue prácticamente similares.

CONCLUSIONES

Basándose en la literatura recopilada de diversas fuentes, se ha llegado a la conclusión de que la utilidad de los protocolos de comunicación IoT es contextual. En el marco de esta investigación, a través de una evaluación exhaustiva de diferentes protocolos de comunicación IoT, se identificó que MQTT es particularmente eficiente y apto para aplicaciones de RV pues, se evidenció su eficacia en la transmisión de datos en tiempo real, gracias a sus cualidades de ligereza, eficiencia energética y aptitud para manejar interacciones en tiempo real.

Como resultado de la investigación se concluye que el protocolo de comunicación MQTT no está limitado al uso netamente en dispositivos IoT como ESP32, Arduino, Raspberry, entre otros, sino también como mensajería de software complejo como los juegos o el control industrial.

En el desarrollo del diseño de un escenario de red, ha existido la necesidad de incorporar otros componentes para garantizar el envío de datos, como se evidenció en esta investigación. En este sentido, se tuvo que agregar el componente Node.js al escenario con MQTT, permitiendo su función como gestor de solicitudes, almacenador de datos en la base de datos y ejecutor de funciones para el envío de datos.

Realizar movimientos en tiempo real mediante HTTP resultó ser inviable, lo que llevó a la implementación del protocolo de comunicación denominado WebSocket para abordar esta problemática. Este protocolo presenta similitudes notables con el comportamiento de MQTT.

El desarrollar todo el sistema desde cero, incluyendo el desarrollo de la aplicación de RV, permitió realizar el cambio de protocolo dentro de la programación de todos los elementos de la red. Si se deseaba tomar un experimento realizado por otro investigador, dicha transición podría enfrentar dificultades significativas o, en última instancia, no ser viable.

Acorde al análisis de los parámetros de rendimiento, se observó que MQTT tiene parámetros de rendimiento mejores que HTTP. Esto se ve reflejado tanto en los tiempo de RTT, en el ancho de banda y en el porcentaje de errores, también se ve reflejado en el uso de CPU y RAM del servidor. En general, usando MQTT para enviar datos puntuales es mejor que HTTP en un 42% para el uso de ancho de banda; un 2.61% en tiempos de RTT; tiene porcentajes de error cercanos a cero a diferencia de HTTP que tiene un porcentaje de error del 2% de errores.

En cuanto a las medidas de los parámetros cuando interactúan en tiempo real, se observó que WebSocket tiene parámetros de rendimiento ligeramente mejores que MQTT para acceder a

servicios de RV. La diferencia entre ambos es que el ancho utilizado con WebSocket es un 14% mejor que MQTT. Pero en porcentaje de errores WebSocket tiene más porcentaje de errores de transmisión. Mientras que los tiempos de ida y vuelta tienen casi los mismos valores promedio de 105ms con un margen de error de 10ms en ambos escenarios.

En relación con la infraestructura de IoT, se ha llegado a la conclusión de que no es indispensable contar con una infraestructura física de gran envergadura para la implementación de protocolos de comunicación para Internet de las cosas. En esta investigación se usó un servidor gratuito en la nube con mínimas capacidades para acceder a servicios de realidad virtual. Asimismo también se utilizó un plan de internet económico o básico que está al alcance de cualquier persona y funcionó tanto para el escenario MQTT como para HTTP. Lo que demuestra que no es necesario contar con una capacidad considerable o desplegar un servidor exclusivo para establecer el sistema. Sin embargo, es importante considerar que en escenarios diferentes al analizado en esta investigación, las capacidades del servidor empleado en este proyecto podrían resultar insuficientes.

Dada la experiencia obtenida con el presente trabajo de investigación se ha concluido que la mejor opción para construir un sistema de acceso a servicios de realidad virtual es la adopción de una diversidad de protocolos, ya sean específicos para IoT o no, con el propósito de aprovechar favorablemente las cualidades únicas de cada uno. Como se ve en este trabajo, se tuvo que cambiar del protocolo HTTP al protocolo WebSocket para lograr una interacción en tiempo real. Sin embargo, colocar los protocolos HTTP, MQTT y WebSocket en un mismo sistema pudiera mejorar por mucho la eficiencia del sistema.

La implementación demostrada en este proyecto abre la posibilidad de mejoras futuras mediante la incorporación de medidas de seguridad avanzadas, técnicas de encriptación y otros mecanismos para mitigar posibles vulnerabilidades del sistema, entre otras consideraciones para elevar o fortalecer una comunicación más segura y robusta

RECOMENDACIONES

Se recomienda que:

Al experimentar con los posibles elementos de una red, se debe verificar el sitio oficial y la documentación de ese sitio ya que existe información comprobada de lo que uno como desarrollador está buscando realizar. Además de esto, también se deben probar estos elementos uno a uno para saber cómo trabajan o cómo se comportan dentro de un ambiente simulado.

Antes de implementar una red, es necesario probar los elementos en una red local lo más parecida a la real para que no existan errores o equivocaciones al momento de programar. Tal como ocurrió en este trabajo de investigación, que en un principio se decidió usar el servidor en la nube y hubo en primer lugar errores de compatibilidad y de programación lo cual requirió un reinicio del servidor, y en segundo lugar hubo ataques a la seguridad al servidor por parte de intrusos que encriptaron toda la información para secuestrarla y pedir una recompensa por esta información.

Este proyecto de investigación ha arrojado resultados satisfactorios al demostrar la viabilidad de emplear un protocolo de comunicación IoT en entornos 3D en comparación con un escenario convencional. Este éxito no solo constituye un logro en sí mismo, sino que también puede servir de referencia valiosa para futuros proyectos similares. En particular, al considerar la eficiencia y la capacidad demostrada por MQTT en entornos de Realidad Virtual (RV), se recomienda a desarrolladores y empresas que contemplen la adopción de este protocolo o alternativas similares en sus próximas implementaciones. Esta sugerencia se basa en la premisa de que dicha elección contribuirá significativamente a lograr una comunicación más fluida y eficiente.

Con la centralización de datos y la naturaleza sensible de las interacciones en tiempo real en RV, es crucial implementar medidas de seguridad robustas para proteger la integridad y privacidad de los datos de los usuarios.

Si bien se ha trabajado con un número específico de usuarios en este escenario, se sugiere llevar a cabo pruebas del sistema en diversas condiciones y con distintos números de usuarios. Este enfoque permitirá obtener una comprensión más completa del rendimiento del sistema en una variedad de situaciones, asegurando así una evaluación más exhaustiva y confiable de su eficacia y adaptabilidad.

BIBLIOGRAFÍA

ACENS. . "Base de datos NoSQL. Qué son y tipos que nos podemos encontrar". *Acens.ComAcens.Com*, (2014), (S.l.) 7 p.

AL-FUQAHA, A. et al. . "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications". *IEEE Communications Surveys and TutorialsIEEE Communications Surveys and Tutorials*, vol. 17, nº 4 (2015), (S.l.) 2347-2376 p.

ALI, A. et al. . "Underwater wireless-to-plastic optical fiber communication systems with a passive front end". *2019 18th International Conference on Optical Communications and Networks, ICOCN 20192019 18th International Conference on Optical Communications and Networks, ICOCN 2019*, (2019), (S.l.) 1-3 p.

ARCOTEL. 2019. "Servicio de acceso a Internet: Boletin estadístico Mayo 2019". . Ecuador: Agencia de Regulación y Control de las Telecomunicaciones - ARCOTEL.

ASHTON, K., *That 'Internet of Things' Thing [en línea],* [blog].1 de julio de 2009. [consulta: 27 junio 2020]. Disponible en: https://www.rfidjournal.com/that-internet-of-things-thing-2

ATZORI, L. et al. . "The Internet of Things: A survey". *Computer NetworksComputer Networks*, vol. 54, nº 15 (2010), (S.l.) 2787-2805 p.

AZNAR LÓPEZ, A. *La red Internet: el modelo TCP/IP [en línea],* [en línea].Madrid, España: Grupo Abantos Formación y Consultoría, 2005, [consulta: 13 agosto 2023]Disponible en: https://elibro.net/es/ereader/espoch/60148?page=22>

BANDYOPADHYAY, D.; & SEN, J. . "Internet of Things: Applications and Challenges in Technology and Standardization". *Wireless Personal CommunicationsWireless Personal Communications*, vol. 58, nº 1 (2011), (S.l.) 49-69 p.

BARRIO, M. *Internet de las cosas [en línea],* [en línea].2ª ed. Madrid, España: Editorial Reus, 2020, [consulta: 14 agosto 2023]Disponible en: https://elibro.net/es/ereader/espoch/185096?page=24>

BEHAL, A. et al. 2023. "Comparing HTTP And COAP For IoT Low-power and Lossy Networks Using The Cooja Simulator". *2023 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*. S.l.: s.n., pp. 1-4. DOI 10.1109/SCEECS57921.2023.10062975.

BLANK, A.G. *TCP/IP Foundations [en línea],* [en línea].USA: Wiley, 2004, [consulta: 13 agosto 2023]Disponible en: https://elibro.net/es/ereader/espoch/180415?page=41>

BONOMI, F. et al. *Fog Computing and Its Role in the Internet of Things [en línea],* [en línea].Helsinki, Finland.: Association for Computing Machinery, 2012, [consulta: 14 mayo 2020]Disponible en: https://doi.org/10.1145/2342509.2342513>

BOTTA, A. et al. . "Integration of Cloud computing and Internet of Things: A survey". *Future Generation Computer SystemsFuture Generation Computer Systems*, vol. 56, (2016), (S.l.) 684-700 p.

CEDIA, *Tecnologías inmersivas multi-usuario orientadas a sistemas sinérgicos de enseñanza-aprendizaje [en línea],* [blog].28 de diciembre de 2020. [consulta: 6 julio 2023]. Disponible en: https://cedia.edu.ec/tecnologias-inmersivas-multi-usuario-orientadas-a-sistemas-sinergicos-de-ensenanza-aprendizaje/

CISCO, *Introducción a Internet de las cosas [en línea],* [blog].22 de junio de 2020. [consulta: 22 junio 2020]. Disponible en: https://static-course-assets.s3.amazonaws.com/I2IoT20/es/index.html#1.2.1.1

CITRIX, *¿Qué es la virtualización? - Definición de virtualización [en línea],* [blog].7 de abril de 2020. [consulta: 4 junio 2020]. Disponible en: https://www.citrix.com/es-mx/glossary/what-is-virtualization.html

CONNOLLY, T.; & BEGG, C. *Database Systems: A Practical Approach to Design, Implementation, and Management, Global Edition*. 6ª ed. S.l.: Pearson Education, 2015, ISBN 978-1-292-06184-9, 1440 p.

COOGAN, C.G.; & HE, B. . "Brain-Computer Interface Control in a Virtual Reality Environment and Applications for the Internet of Things". *IEEE AccessIEEE Access*, vol. 6, (2018), (S.l.) 10840-10849 p.

CÓRDOVA, F. 2008. "TECNOLOGIAS DE ACCESO". . S.l.: imaginar.org.

CORSARO, A. . "A Comparative Study of Data-Sharing Standards for the Internet of Things". *The Journal of Information Technology Management Cutter IT JournalThe Journal of Information Technology Management Cutter IT Journal*, vol. 27, nº 11 (2014), (S.l.) 23-29 p.

COUCH, L.W. *Digital and Analog Communication*. 8thª ed. 1 Lake Street, Upper Saddle River, New Jersey, 07458: Prentice Hall, 2013, ISBN 9780132915380, 789 p.

DA SILVA, M.M. *Introduction to Data Communications and Networking [en línea],* [en línea].Boca Raton: CRC Press, 2018, Disponible en: https://doi.org/10.1201/9781315368658>

DAY, J.D.; & ZIMMERMANN, H. . "The OSI reference model". *Proceedings of the IEEEProceedings of the IEEE*, vol. 71, nº 12 (1983), (S.l.) 1334-1340 p.

DE BEER, A.S. et al. . "Contactless Power Line Communications at 2.45GHz". *2016 International Symposium on Power Line Communications and its Applications, ISPLC 20162016 International Symposium on Power Line Communications and its Applications, ISPLC 2016*, (2016), (S.l.) 42-45 p.

DORDAL, P.L. *An Introduction to Computer Networks [en línea],* [en línea].2ª ed. llinois-USA: s.n., 2020, Disponible en: licensed>

DUEÑAS NOGUERAS, J. *Sistemas de información y bases de datos: COMT0110 [en línea],* [en línea].Málaga, España: IC Editorial, 2023, [consulta: 13 agosto 2023]Disponible en: https://elibro.net/es/ereader/espoch/229144?page=192>

EMQ, *About EMQ [en línea],* [blog].2022. Disponible en: https://www.emqx.com/en/about

EPIC GAMES INC, *Unreal Engine [en línea],* [blog].2022. Disponible en: https://www.unrealengine.com

ESCALONA, A.S. *Tecnologías de acceso para las icts. El instalador, los servicios y las redes [en línea],* [en línea].S.l.: Ediciones Experiencia, 2012, [consulta: 25 junio 2020]Disponible en: https://books.google.com.ec/books?id=av-LDwAAQBAJ&pg=PA16&dq=tecnologias+de+acceso&hl=es&sa=X&ved=2ahUKEwjH-eTSgp7qAhVlZTUKHb1VBfkQ6AEwAHoECAAQAg#v=onepage&q=cobre&f=false>

FANG, W.D. et al. . "Research on the application-driven architecture in internet of things". *2010 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE)2010 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE)*, vol. 293, (2016), (Chengdu, China) 458-465 p.

FERNEY, W. et al. . "Tecnología internet of things (IoT) y el big data Internet of Thing (IoT) technology and the big data". *Mare Ingenii IngenieríasMare Ingenii Ingenierías*, vol. 1, (2019), (S.l.) 76 p.

FIELDING, R. et al. 1999. "Hypertext Transfer Protocol -- HTTP/1.1". . S.l.:

FIGUEIRAS VIDAL, A.R. *Una panorámica de las telecomunicaciones [en línea],* [en línea].Madrid, España: Prentice Hall, 2002, [consulta: 29 abril 2020]Disponible en: https://books.google.com.gt/books?id=mCVSS4LVuO8C&printsec=copyright#v=onepage&q&f=false>

FITZGERALD, J. et al. *Business Data Communications And Networking*. 11th[a] ed. S.l.: s.n., 2011, ISBN 111808683X,

FOROUZAN, B.A. *Data communications and networking*. New York, NY: McGraw-Hill, 2012, ISBN 978-0-07-337622-6,

GEMIRTER, C.B. et al. . "A Comparative Evaluation of AMQP, MQTT and HTTP Protocols Using Real-Time Public Smart City Data". En: ISSN: 2521-1641, *2021 6th International Conference on Computer Science and Engineering (UBMK)2021 6th International Conference on Computer Science and Engineering (UBMK)*, (2021), (S.l.) 542-547 p.

GENG, H. *Internet of Things and Data Analytics*. Hoboken, New Jersey: John Wiley & Sons Inc, 2017, ISBN 9781119173632, 1178 p.

GONZÁLEZ GARCÍA, A.J., *IoT: Dispositivos, tecnologías de transporte y aplicaciones* (Trabajo de titulación) (maestría o doctoral). Universitat Oberta de CatalunyaEspaña11 de junio de 2017.

GONZÁLEZ, M.S., *La última milla, tecnologías de acceso [en línea],* [blog].5 de noviembre de 2012. [consulta: 24 junio 2020]. Disponible en: http://redestelematicas.com/la-ultima-milla/

GONZÁLEZ RIO, M.D. *Tecnologías de Virtualización [en línea],* [en línea].S.l.: Createspace Independent Pub, 2014, Disponible en: https://books.google.com.ec/books?id=-ZzfCgAAQBAJ>

HARRINGTON, J.L. *Relational Database Design and Implementation - 4th Edition [en línea],* [en línea].4[a] ed. S.l.: Elsevier, 2016, [consulta: 13 agosto 2023]Disponible en: https://shop.elsevier.com/books/relational-database-design-and-implementation/harrington/978-0-12-804399-8>

HERNÁNDEZ-SAMPIERI, R.; & MENDOZA, C. *Metodología de la investigación. Las rutas cuantitativa, cualitativa y mixta [en línea],* [en línea].7[a] ed. Ciudad de México, México: Mc Graw Hill Education, 2018, [consulta: 25 agosto 2023]Disponible en: https://virtual.cuautitlan.unam.mx/rudics/?p=2612>

HOU, X. et al. . "IIoT-MEC: A Novel Mobile Edge Computing Framework for 5G-enabled IIoT". *2019 IEEE Wireless Communications and Networking Conference (WCNC)* [en línea], 2019, (S.l.) ISSN 1558-2612. DOI 10.1109/WCNC.2019.8885703. Disponible en: https://ieeexplore.ieee.org/document/8885703

HU, X. et al. . "Impulsive Noise Cancellation for MIMO Power Line Communications". *Journal of CommunicationsJournal of Communications*, vol. 9, nº 3 (2014), (S.l.) 241-247 p.

IBM CORPORATION, *Components of URL [en línea]*, [blog].2015a. Disponible en: https://www.ibm.com/docs/en/cics-ts/5.3?topic=concepts-components-url#dfhtl_uricomp

IBM CORPORATION, *Modelo de comunicación de publicación/suscripción [en línea]*, [blog].2015b. Disponible en: https://www.ibm.com/docs/es/iis/11.5?topic=SSZJPZ_11.5.0/com.ibm.swg.im.iis.conn.wsmq.stages.doc/topics/c_cwsmqcon_Publish_Subscribe_Comm_Model.html

IBM CORPORATION, *Status codes and reason phrases [en línea]*, [blog].2015c. Disponible en: https://www.ibm.com/docs/en/cics-ts/5.3?topic=concepts-status-codes-reason-phrases

IBM CORPORATION, *The HTTP protocol [en línea]*, [blog].2015d. Disponible en: https://www.ibm.com/docs/en/cics-ts/5.3?topic=concepts-http-protocol

JEYARAMAN, R. et al. 2012. "OASIS Advanced Message Queuing Protocol (AMQP) Version 1.0 Specification URIs Chairs: Editors". *OASIS Advanced Message Queuing Protocol*. S.l.: AMQP) TC.

JOYANES, L. *Internet de las cosas [en línea]*, [en línea].Colombia: Alpha Editorial, 2021, [consulta: 14 agosto 2023]Disponible en: https://www.alphaeditorialcloud.com/reader/9789587786934-compressed-1611871601>

KUROSE, J.F.; & ROSS, K.W. *Redes de computadoras: un enfoque descendente*. 7ª ed. Boston: Pearson Educación, 2017, ISBN 978-84-9035-528-2, 704 p.

KUSNETZKY, D. *Virtualization: A Manager's Guide*. S.l.: O'Reilly Media, Inc., 2011, ISBN 978-1-4493-0645-8, 73 p.

LAN, K.-C. et al. . "Slow Breathing Exercise with Multimodal Virtual Reality: A Feasibility Study". *SensorsSensors*, vol. 21, nº 16 (2021), (S.l.) 5462 p.

LANDINGER, T.F. et al. . "Power Line Communications in Automotive Traction Batteries: A Proof of Concept". (2020), (S.l.) 1-5 p.

LEE, I.; & LEE, K. . "The Internet of Things (IoT): Applications, investments, and challenges for enterprises". *Business HorizonsBusiness Horizons*, vol. 58, nº 4 (2015), (S.l.) 431-440 p.

LÓPEZ QUEROL, J. et al. . "Algoritmia y bases de datos". [en línea], 2023, (S.l.) [consulta: 13 agosto 2023]. Disponible en: https://elibro.net/es/ereader/espoch/230563

MARKETING PUBLISHING. *Toma de decisiones eficaces*. S.l.: Ediciones Díaz de Santos, 2012, ISBN 978-84-9969-172-5, 284 p.

MARQUÉS, M. *Bases de Datos*. 1eraª ed. S.l.: Universitat Jaume, 2011a, ISBN 9788469301463,

MARSIC, I. *Computer Networks: Performance and Quality of Service [en línea],* [en línea].New Jersey - USA: Rutgers University, 2013, [consulta: 13 agosto 2023]Disponible en: http://www.ece.rutgers.edu/~marsic/books/CN/>

MCMILLAN, T. *Cisco Networking Essentials*. S.l.: John Wiley & Sons, 2015, ISBN 978-1-119-09212-4, 482 p.

MELL, P.; & GRANCE, T. . "The NIST Definition of Cloud Computing". [en línea], 2011, (Gaithersburg) [consulta: 14 mayo 2020]. DOI 10.6028/NIST.SP.800-145. Disponible en: https://www.nist.gov/publications/nist-definition-cloud-computing

MICHENG, Z.; & LIPING, Z. . "Unity3D-based Visual Field Obstruction Analysis and Simulation of Sensor". *Journal of Physics: Conference SeriesJournal of Physics: Conference Series*, vol. 1617, nº 1 (2020), (S.l.) 012051 p.

MINISTERIO DE TELECOMUNICACIONES Y DE LA SOCIEDAD DE LA INFORMACIÓN, *¿Sabe para qué sirve la fibra óptica? [en línea],* [blog].20 de enero de 2017. [consulta: 25 junio 2020]. Disponible en: https://www.telecomunicaciones.gob.ec/sabe-para-que-sirve-la-fibra-optica/

NAVATHE, S.B.; & ELMASRI, R. *Fundamentos de Sistemas de Bases de Datos*. 5taª ed. S.l.: Pearson, 2007, ISBN 9788478290857,

OASIS. 2019. "MQTT Version 5.0 OASIS Standard". *MQTT.org*. S.l.:

OBJECT MANAGEMENT GROUP, *What is DDS? [en línea],* [blog].2021. Disponible en: https://www.dds-foundation.org/what-is-dds-3/

PATTERSON, D.; & HENNESSY, J. *Estructura y diseño de computadores [en línea],* [en línea].4ª ed. Barcelona, España: REVERTÉ, 2011, [consulta: 9 agosto 2023]Disponible en: https://www.alphaeditorialcloud.com/reader/estructura-y-diseno-de-computadores>

PESSOLANI, P. et al. 2012. "Sistema de Virtualización con Recursos Distribuidos". *XIV Workshop de Investigadores en Ciencias de la Computación* [en línea], . S.l.: Red de Universidades con Carreras en Informática (RedUNCI), pp. 49-53. [consulta: 10 junio 2020]. ISBN 978-950-766-082-5. Disponible en: http://sedici.unlp.edu.ar/handle/10915/18375.

RADIANTI, J. et al. . "A systematic review of immersive virtual reality applications for higher education: Design elements, lessons learned, and research agenda". *Computers & EducationComputers & Education*, vol. 147, (2020), (S.l.) 103778 p.

RAYA CABRERA, J.L. *Sistemas operativos en red [en línea],* [en línea].S.l.: RA-MA Editorial, 2015, [consulta: 13 agosto 2023]Disponible en: https://elibro.net/es/ereader/espoch/62454>

RED HAT, *La virtualización [en línea],* [blog].29 de agosto de 2018. [consulta: 1 junio 2020]. Disponible en: https://www.redhat.com/es/topics/virtualization

REGALADO JALCA, J.J. et al. *Redes de computadoras [en línea],* [en línea].1ª ed. S.l.: Editorial Científica 3Ciencias, 2018, [consulta: 10 agosto 2023]Disponible en: http://dx.doi.org/10.17993/IngyTec.2018.32>

ROSE, K. et al. *La Internet de las cosas - Una breve reseña [en línea],* [en línea].USA: Internet Society (ISOC), 2015, [consulta: 27 abril 2020]Disponible en: www.internetsociety.org/iot>

SADALAGE, P.J.; & FOWLER, M. *NoSQL distilled: a brief guide to the emerging world of polyglot persistence*. Upper Saddle River, New Jersey: Addison-Wesley, 2013, ISBN 978-0-321-82662-6, 164 p. QA76.9.D32 S228 2013

SÁNCHEZ RUBIO, M. et al. *Redes de computadores [en línea],* [en línea].S.l.: Editorial Universidad de Alcalá, 2020, [consulta: 9 agosto 2023]Disponible en: https://elibro.net/es/ereader/espoch/131606>

SEPÚLVEDA RODRÍGUEZ, L.E. et al. . "A Survey of Virtualization Technologies: Towards a New Taxonomic Proposal". *Ingeniería e Investigación* [en línea], 2022, (S.l.) 42 (3), [consulta: 14 agosto 2023]. ISSN 0120-5609. DOI 10.15446/ing.investig.97363. Disponible en: http://www.scielo.org.co/scielo.php?script=sci_abstract&pid=S0120-56092022000300214&lng=en&nrm=iso&tlng=en

SIMISCUKA, A.A. et al. . "Real-Virtual World Device Synchronization in a Cloud-Enabled Social Virtual Reality IoT Network". *IEEE AccessIEEE Access*, vol. 7, (2019), (S.l.) 106588-106599 p.

SINGH, J.; & KUMAR, R. . "A Review Paper on Networking Topologies". vol. 5, nº 9 (2018), (S.l.)

SINGH, N.; & SINGH, S. . "Virtual reality: A brief survey". En: editorial. Institute of Electrical and Electronics Engineers Inc., *2017 International Conference on Information Communication and Embedded Systems, ICICES 2017* [en línea], 2017, (S.l.) [consulta: 29 junio 2020]. DOI 10.1109/ICICES.2017.8070720. Disponible en: https://ieeexplore.ieee.org/document/8070720

SMITH, J.E.; & NAIR, R. . "The architecture of virtual machines". *ComputerComputer*, vol. 38, nº 5 (2005), (S.l.) 32-38 p.

STALLINGS, W. *Operating systems : internals and design principles*. 7thª ed. 1 Lake Street, Upper Saddle River, New Jersey, 07458: Prentice Hall, 2012, ISBN 9780132309981,

STALLINGS, W. *Data and computer communications*. 10thª ed. 1 Lake Street, Upper Saddle River, New Jersey, 07458: Prentice Hall, 2014, ISBN 9780133506488, 919 p.

TANENBAUM, A.S.; & WETHERALL, D.J. *Redes de computadoras*. 5ª ed. México: Pearson Education, 2012, ISBN 978-607-32-0817-8, 102 p.

TELEFÓNICA I+D. *Las Telecomunicaciones de Nueva Generación [en línea]*, [en línea].Barcelona, España: Lerko print S.A, 2002, Disponible en: telefonica.com>

TSCHOFENIG, H. et al. . "RFC 7452 - Architectural Considerations in Smart Object Networking". *IETF* [en línea], 2015, (S.l.) [consulta: 27 abril 2020]. ISSN 2070-1721. Disponible en: http://trustee.ietf.org/license-info

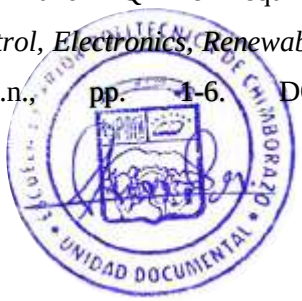
UNITY, *Unity Platform [en línea]*, [blog].2022. Disponible en: https://unity.com/products/unity-platform

UNIVERSITY OF SOUTH FLORIDA, *Chapter 1: What is a Network? [en línea]*, [blog].2013. [consulta: 9 agosto 2023]. Disponible en: https://fcit.usf.edu/network/chap1/chap1.htm

VILLAR FERNÁNDEZ, E.E., & GÓMEZ, J., *Virtualización de servidores de telefonía IP en GNU/Linux* [en línea], (Trabajo de titulación) (maestría o doctoral). [en línea] Universidad de AlmeríaAlmeríajunio de 2010. [Consulta: 31-mayo-2020]. Disponible en: http://www.adminso.es

VMWARE, *Virtualización [en línea]*, [blog].10 de abril de 2020. [consulta: 2 junio 2020]. Disponible en: https://www.vmware.com/latam/solutions/virtualization.html

YOKOTANI, T., & SASAKI, Y. 2016. "Comparison with HTTP and MQTT on required network resources for IoT". *2016 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC)*. S.l.: s.n., pp. 1-6. DOI 10.1109/ICCEREC.2016.7814989.



ANEXOS

ANEXO A: PROGRAMACIÓN EN NODE.JS PARA GESTIÓN DE LA APLICACIÓN CON PROTOCOLO MQTT

server.js

```
const MongoClient = require('mongodb').MongoClient;
const mqtt = require('mqtt');

let c=0;
let query={};
let newvalue={};
const client_mongo = new MongoClient('mongodb://localhost', {useNewUrlParser: true});
client_mongo.connect((err) => {
    if (err) throw err;
    console.log("Connected to MongoDB");
});



const client_mqtt = mqtt.connect('mqtt://localhost');
client_mqtt.on('connect', () => {
  console.log('Connected to MQTT broker');
});

client_mqtt.subscribe(['datos','datos2', 'reset','intentosA','get','get2', 'reset2','intentosB']);

client_mqtt.on('message', (topic, message) => {

  //console.log(`Received message on topic ${topic}: ${message}`);
  if(topic == 'reset'){
    var collection = client_mongo.db("mqtt").collection("Control");
    const  ObjectID = require('mongodb').ObjectId;
    query = {_id: ObjectID("63c767eef970515b1b8b8b47")}; // the field and value you want to find
    newValue = { $set: { Intentos: message.toString()} }; // the new value you want to
    console.log("Nuevo Valor:");
    console.log(newValue);
    collection.updateOne(query, newValue, (err, res) => {
    if (err) throw err;
    console.log("Intentos Brazo 1 actualizado");
  });
  }
   if(topic == 'reset2'){
    var collection = client_mongo.db("mqtt").collection("Control");
    const  ObjectID = require('mongodb').ObjectId;
    query = {_id: ObjectID("640d19a5c4a862fcb03b95f6")}; // the field and value you want to find
    newValue = { $set: { Intentos: message.toString()} }; // the new value you want to
    console.log("Nuevo Valor:");
```

```
    console.log(newValue);
    collection.updateOne(query, newValue, (err, res) => {
    if (err) throw err;
    console.log("Intentos Brazo 2 actualizado");
  });
  }


  if(topic == 'intentosA'){
    var collection = client_mongo.db("mqtt").collection("Control");
    const  ObjectID = require('mongodb').ObjectId;
    //const query0 = {_id: ObjectID("63c767eef970515b1b8b8b47")};
    const query0 = {_id: ObjectID(message.toString())};
    const projection0 = { Intentos: 1, _id: 0 };
    collection.findOne(query0,projection0, (err, document) => {
    if (err) throw err;
    console.log(document);
    //const jsonString = JSON.stringify(document);
    c=document['Intentos'];
    console.log(document['Intentos']);
    client_mqtt.publish('Intentos', document['Intentos']);
    console.log("publicado Valor Intento 1");

  });
  }
  if(topic == 'intentosB'){
    var collection = client_mongo.db("mqtt").collection("Control");
    const  ObjectID = require('mongodb').ObjectId;
    //const query0 = {_id: ObjectID("63c767eef970515b1b8b8b47")};
    const query0 = {_id: ObjectID(message.toString())};
    const projection0 = { Intentos: 1, _id: 0 };
    collection.findOne(query0,projection0, (err, document) => {
    if (err) throw err;
    console.log(document);
    //const jsonString = JSON.stringify(document);
    c=document['Intentos'];
    console.log(document['Intentos']);
    client_mqtt.publish('Intentos2', document['Intentos']);
    console.log("publicado Valor Intento 2");

  });
  }
```

```
85
86
87
88    if(topic == 'datos'){
89      var collection = client_mongo.db("mqtt").collection("Datos");
90      var jsonData=JSON.parse(message.toString());
91      const query2 =  { intento: jsonData.intento };
92      const newData = { $set: jsonData };
93      collection.findOneAndUpdate(query2,newData, function(err, res) {
94          console.log(res);
95      });
96       /*const myobj=message.toString();
97
98      collection.insertOne(myobj, function(err, res) {
99          console.log("1 document inserted");
100     });*/
101
102   }
103
104   if(topic == 'datos2'){
105     var collection = client_mongo.db("mqtt").collection("Datos2");
106     var jsonData=JSON.parse(message.toString());
107     const query2 =  { intento: jsonData.intento };
108     const newData = { $set: jsonData };
109     collection.findOneAndUpdate(query2,newData, function(err, res) {
110         console.log(res);
111     });
112      /*const myobj=message.toString();
113
114     collection.insertOne(myobj, function(err, res) {
115         console.log("1 document inserted");
116     });*/
117
118   }
119   if(topic == 'get'){
120     var collection = client_mongo.db("mqtt").collection("Datos");
121     const  ObjectID = require('mongodb').ObjectId;
122     const query0 = {_id: ObjectID(message.toString())};
123      //const projection0 = { Intentos: 1, _id: 0 };
124     collection.findOne(query0, (err, document) => {
125     if (err) throw err;
126     console.log(document);
127      //const jsonString = JSON.stringify(document);
128      //c=document['Intentos'];
129      //console.log(document['Intentos']);
130     client_mqtt.publish('dataOut', JSON.stringify(document));
131     console.log("publicado");
132     });
```

```
135     if(topic == 'get2'){
136     var collection = client_mongo.db("mqtt").collection("Datos2");
137     const  ObjectID = require('mongodb').ObjectId;
138     const query0 = {_id: ObjectID(message.toString())};
139      //const projection0 = { Intentos: 1, _id: 0 };
140     collection.findOne(query0, (err, document) => {
141     if (err) throw err;
142     console.log(document);
143      //const jsonString = JSON.stringify(document);
144      //c=document['Intentos'];
145      //console.log(document['Intentos']);
146     client_mqtt.publish('dataOut2', JSON.stringify(document));
147     console.log("publicado");
148     });
149   }
150
151 });
```

ANEXO B: PROGRAMACIÓN EN NODE.JS PARA GESTIÓN DE LA APLICACIÓN CON PROTOCOLO HTTP

server.js

```
var express = require('express');
var bodyParser = require('body-parser');
var mongodb = require('mongodb');
var MongoClient = mongodb.MongoClient;
var ObjectId = mongodb.ObjectId;

var app = express();

app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: true }));

var url = 'mongodb://localhost:27017/http';

MongoClient.connect(url, { useUnifiedTopology: true }, function(err, client) {
    if (err) throw err;

    console.log("¡Conectado a la base de datos!");

    app.post('/data', function(req, res) {
        var collectionName = req.body.collection;
        var db = client.db('http');
        var collection = db.collection(collectionName);
        var id = new ObjectId(req.body.id);
        var changes = req.body.changes;

        var updateObject = { $set: {} };
        for (var field in changes) {
            if (changes.hasOwnProperty(field)) {
                updateObject.$set[field] = changes[field];
            }
        }

        collection.updateOne({ _id: id }, updateObject, function(err, result) {
            if (err) throw err;

            if (result.matchedCount === 0) {
                res.status(404).send("El documento no fue encontrado");
            } else {
                res.send("Documento actualizado exitosamente");
                console.log(updateObject);
            }
        });
    });

    app.post('/control', function(req, res) {
        var collectionName = req.body.collection;
        var db = client.db('http');
        var collection = db.collection(collectionName);
```

```
        var id = new ObjectId(req.body.id);
        var field = req.body.field;
        var newValue = req.body.newValue;
        var updateObject = { $set: {} };
        updateObject.$set[field] = newValue;

        collection.updateOne({ _id: id }, updateObject, function(err, result) {
            if (err) throw err;

            if (result.matchedCount === 0) {
                res.status(404).send("El documento no fue encontrado");
            } else {
                res.send("Documento actualizado exitosamente");
                console.log(updateObject);
            }
        });
    });

    app.get('/data', function(req, res) {
        var collectionName = req.query.collection;
        var db = client.db('http');
        var collection = db.collection(collectionName);
        var id = new ObjectId(req.query.id);

        collection.findOne({ _id: id }, function(err, item) {
            if (err) throw err;

            if (item === null) {
                res.status(404).send("El documento no fue encontrado");
            } else {
                res.send(item);
            }
        });
    });

    app.listen(3000, function() {
        console.log('Aplicación escuchando en el puerto 3000');
    });
});
```

ANEXO C: PROGRAMACIÓN EN UNITYS PARA GESTIÓN DE LA APLICACIÓN CON PROTOCOLO MQTT CONEXIÓN Y MANEJO DE VARIABLES

Conexion.cs*

Archivos varios — Conexion

```
using System.Collections;
using UnityEngine;
using UnityEngine.UI;
using System.Text;
using UnityEngine.SceneManagement;
using uPLibrary.Networking.M2Mqtt;
using uPLibrary.Networking.M2Mqtt.Messages;
using System;
using System.Globalization;

public class Conexion : MonoBehaviour
{
    /*public Text usuario;
    public Text passw;*/
    /*public Text user;
    public InputField pwd;*/
    public GameObject Error;
    public GameObject OK;
    public GameObject uno;
    public GameObject dos;
    // MQTT
    public int brokerPort = 1883;
    public int estudiante = 0;
    public string hubAddress;
    public string deviceId;
    public string password="12345";
    public string username= "Estudiante";
    public MqttClient client;
    public static Conexion ConexionA;
    Slider mySlider;


    private void Awake()
    {
        if (ConexionA == null)
        {
            GameObject[] objs = GameObject.FindGameObjectsWithTag("Com");
            if (objs.Length > 1)
            {
                Destroy(this.gameObject);
            }
            DontDestroyOnLoad(this.gameObject);
        }


    }
```

```
48      public void Start()
49      {
50
51
52      }
53
54      public void Authentication()
55      {
56          StartCoroutine(inicio());
57          estudiante = 1;
58          //Subscribe("Estudiante");
59      }
60
61      public void Authentication2()
62      {
63          StartCoroutine(inicio());
64          estudiante = 2;
65          //Subscribe("Estudiante");
66      }
67
68
69      public IEnumerator inicio()
70      {
71
72          /*deviceId = this.user.text;
73          username = this.user.text;
74          password = this.pwd.text;*/
75          /*X509Certificate cert = new X509Certificate();
76          cert.Import(certificate.bytes);
77          Debug.Log("Using the certificate '" + cert + "'");
78
79          this.client = new MqttClient(
80              brokerHostName: this.hubAddress, brokerPort, true, cert, null, MqttSslProtocols.TLSv1_0, MyRemote
81          );*/
82
83          this.client = new MqttClient(
84              brokerHostName: this.hubAddress
85          );
86
87          Debug.Log(username);
88          Debug.Log(password);
89
90          this.client.Connect(
91
92              clientId: this.deviceId.ToString(),
93              username: this.username.ToString(),
```

```
            password: this.password.ToString()

        );


        if (this.client.IsConnected)
        {
            Debug.Log("Conectado:" + this.client.IsConnected);
            uno.SetActive(false);
            dos.SetActive(false);
            OK.SetActive(true);
            client.Subscribe(new string[] { "#" }, new byte[] { MqttMsgBase.QOS_LEVEL_EXACTLY_ONCE });
            yield return new WaitForSeconds(2);
            //yield return SwitchToVR();
            SceneManager.LoadScene("Aula Estudiantes");

            //mySlider = GameObject.FindWithTag("Slider1").GetComponent<Slider>();
        }
        else
        {
            Error.SetActive(true);
            /*pwd.Select();
            pwd.text = "";*/
            yield return new WaitForSeconds(1);
            Error.SetActive(false);
            /*Text = this.ConexionA.GetComponent<Text>();
            Text.text = "Error al Ingresar los Datos";*/
        }
        //Error al Ingresar los Datos
        ;
    }

    public void Subscribe(string topic)
    {
        client.MqttMsgPublishReceived += this.onReceive;
        client.Subscribe(new string[] { topic }, new byte[] { MqttMsgBase.QOS_LEVEL_AT_MOST_ONCE });
    }

    public void onReceive(object sender, uPLibrary.Networking.M2Mqtt.Messages.MqttMsgPublishEventArgs e)
    {
        int n1,n2, n3, n4;
        string ang1, ang2, ang3;

        string msg = Encoding.UTF8.GetString(e.Message);
        if (e.Topic == "Estudiante")
        {
            Debug.Log(msg);
            n1 = msg.IndexOf("angulo1:");
```

```
142                 n2 = msg.IndexOf("angulo2:");
143                 n3 = msg.IndexOf("angulo3:");
144                 n4 = msg.IndexOf("fin");
145                 ang1=msg.Substring(n1, n2);
146                 Debug.Log(ang1);
147                 ang2 = msg.Substring(n2, n3 - n2);
148                 Debug.Log(ang2);
149                 ang3 = msg.Substring(n3, n4 - n3);
150                 Debug.Log(ang3);
151 
152                 Debug.Log(mySlider.value);
153 
154             }
155         }
156 
157         // Call via `StartCoroutine(SwitchToVR())` from your code. Or, use
158         // `yield SwitchToVR()` if calling from inside another coroutine.
159         IEnumerator SwitchToVR()
160         {
161             // Device names are lowercase, as returned by `XRSettings.supportedDevices`.
162             string desiredDevice = "cardboard"; // Or "cardboard".
163 
164             // Some VR Devices do not support reloading when already active, see
165             // https://docs.unity3d.com/ScriptReference/XR.XRSettings.LoadDeviceByName.html
166             if (string.Compare(UnityEngine.XR.XRSettings.loadedDeviceName, desiredDevice, true) != 0)
167             {
168                 UnityEngine.XR.XRSettings.LoadDeviceByName(desiredDevice);
169 
170                 // Must wait one frame after calling `XRSettings.LoadDeviceByName()`.
171                 yield return null;
172             }
173 
174             // Now it's ok to enable VR mode.
175             UnityEngine.XR.XRSettings.enabled = true;
176         }
177 
178 
179 
180     }
181 
182 
183 
```

ANEXO D: PROGRAMACIÓN EN UNITY PARA GESTIÓN DE LA APLICACIÓN CON PROTOCOLO HTTP CONEXIÓN Y MANEJO DE VARIABLES

HttpClient.cs

Archivos varios — HttpClient

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.Networking;
using System;
using System.Globalization;
using Newtonsoft.Json;
using Newtonsoft.Json.Linq;


public class HttpClient : MonoBehaviour
{

    public static HttpClient instance;
    public int Estudiante;
    public Dropdown myDropdown;
    public Dropdown myDropdown2;
    public GameObject Botones1;
    public GameObject Botones2;
    public Slider mySlider1;
    public Slider mySlider2;
    public Slider mySlider3;
    public Slider mySlider12;
    public Slider mySlider22;
    public Slider mySlider32;
    public Text inte;
    public Text inte2;
    int intento = 0;
    int intento2 = 0;
    public Button Envio;
    public Button Envio2;
    bool flagslider1 = false, flagslider2 = false, flagslider3 = false;
    //bool flagslider12 = false, flagslider22 = false, flagslider32 = false;
    public string dato_slider1;
    public string dato_slider2;
    public string dato_slider3;
    [Serializable]
    public class Sliders
    {
        public string slider1;
        public string slider2;
        public string slider3;
    }

```

```
48
49          if (Estudiante == 1)
50          {
51              CallGetRequest2("Control", "63c767eef970515b1b8b8b47");
52
53              mySlider12.interactable = false;
54              mySlider22.interactable = false;
55              mySlider32.interactable = false;
56
57              mySlider1.onValueChanged.AddListener(delegate { OnValueChanged1(); });
58              mySlider2.onValueChanged.AddListener(delegate { OnValueChanged2(); });
59              mySlider3.onValueChanged.AddListener(delegate { OnValueChanged3(); });
60
61              Botones2.gameObject.SetActive(false);
62
63              myDropdown.onValueChanged.AddListener(delegate
64              {
65                  DropdownValueChanged(myDropdown);
66              });
67
68          }
69
70          if (Estudiante == 2)
71          {
72              //leer intentos
73              mySlider1.interactable = false;
74              mySlider2.interactable = false;
75              mySlider3.interactable = false;
76
77              Botones1.gameObject.SetActive(false);
78              myDropdown2.onValueChanged.AddListener(delegate
79              {
80                  DropdownValueChanged2(myDropdown2);
81              });
82          }
83      }
84
85      private void OnValueChanged1()
86      {
87
88          flagslider1 = true;
89      }
90
91      private void OnValueChanged2()
92      {
93
94          flagslider2 = true;
95      }
96
```

```
97      private void OnValueChanged3()
98      {
99
100         flagslider3 = true;
101     }
102
103     void Update()
104     {
105
106
107         if (KnotPoints.instance.knotPoints < 5)
108         {
109             Envio.gameObject.SetActive(false);
110
111         }
112
113         if (KnotPoints1.instance.knotPoints1 < 5)
114         {
115             Envio2.gameObject.SetActive(false);
116
117         }
118
119         if (intento < 4)
120         {
121             if (KnotPoints.instance.knotPoints == 5)
122             {
123                 Envio.gameObject.SetActive(true);
124
125             }
126
127         }
128
129         if (intento2 < 4)
130         {
131             if (KnotPoints1.instance.knotPoints1 == 5)
132             {
133                 Envio2.gameObject.SetActive(true);
134
135             }
136         }
137
138         if (Input.GetKeyDown(KeyCode.F5)) // Usamos KeyCode.F5 para representar la tecla F5
139         {
140             Actualizar();
141         }
142
143         if (flagslider1 || flagslider2 || flagslider3)
144         {
```

```
            Sliders Slider = new Sliders();
            float sliderValue1 = mySlider1.value;
            float sliderValue2 = mySlider2.value;
            float sliderValue3 = mySlider3.value;
            dato_slider1 = mySlider1.value.ToString();
            dato_slider2 = mySlider2.value.ToString();
            dato_slider3 = mySlider3.value.ToString();

            Slider.slider1 = dato_slider1;
            Slider.slider2 = dato_slider2;
            Slider.slider3 = dato_slider3;

            Debug.Log("Slider1: " + sliderValue1);
            Debug.Log("Slider2: " + sliderValue2);
            Debug.Log("Slider3: " + sliderValue3);

            string jsonSlider = JsonUtility.ToJson(Slider);
            WebSocketClient.instance.SendMessage("user1", jsonSlider);
            flagslider1 = false;
            flagslider2 = false;
            flagslider3 = false;

        }

    }

    public void HTTPbase()
    {
        if (intento < 4)
        {
            if (intento == 1)
            {
                CallPostRequest("Datos", "63c8c3092588e8b1f29547d0");
                CallPostRequest2("Control", "63c767eef970515b1b8b8b47", "Intentos", "2");
            }
            if (intento == 2)
            {
                CallPostRequest("Datos", "63c8c30d2588e8b1f29547d1");
                CallPostRequest2("Control", "63c767eef970515b1b8b8b47", "Intentos", "3");
            }

            if (intento == 3)
            {
                CallPostRequest("Datos", "63c8c3112588e8b1f29547d2");
                CallPostRequest2("Control", "63c767eef970515b1b8b8b47", "Intentos", "4");
```

```
            }
            intento = intento + 1;
            Debug.Log("Numero de Intento Actual Brazo 1: " + intento);
            inte.text = (4 - intento).ToString();
            KnotPoints.instance.ClearPoints();
            KnotPoints.instance.knotPoints = 0;
        }
    }

    public void Actualizar()
    {
        CallGetRequest2("Control", "63c767eef970515b1b8b8b47");
    }


    void DropdownValueChanged(Dropdown change)
    {
        Debug.Log("Selected: " + change.value);
        if (change.value == 0)
        {

        }

        if (change.value == 1)
        {

        }

        if (change.value == 2)
        {

        }

    }

    void DropdownValueChanged2(Dropdown change)
    {
        Debug.Log("Selected: " + change.value);

        if (change.value == 0)
        {

        }
        if (change.value == 1)
        {

        }
        if (change.value == 2)
```

```
238         {
239
240         }
241
242     }
243
244
245     [System.Serializable]
246     public class Intento
247     {
248
249         public string Intentos;
250     }
251
252     [System.Serializable]
253     public class Angulo
254     {
255
256         public Quaternion Angulo1;
257         public Quaternion Angulo2;
258         public Quaternion Angulo3;
259     }
260
261     [System.Serializable]
262     public class Movimiento
263     {
264         public Angulo Movimiento1;
265         public Angulo Movimiento2;
266         public Angulo Movimiento3;
267         public Angulo Movimiento4;
268         public Angulo Movimiento5;
269     }
270
271     [System.Serializable]
272     public class RequestBody
273     {
274         public string collection;
275         public string id;
276         public Movimiento changes;
277     }
278
279     [System.Serializable]
280     public class RequestBody2
281     {
282         public string collection;
283         public string id;
284         public string field;
285         public string newValue;
```

```
286     }
287
288     public Dictionary<string, string> changes = new Dictionary<string, string>();
289
290     public void CallPostRequest(string collection, string id)        //Guardar Datos em datos con ID
291     {
292         Movimiento cambios = new Movimiento();
293         cambios.Movimiento1 = new Angulo();
294         cambios.Movimiento1.Angulo1 = (KnotPoints.instance.theta1Array[0]);
295         cambios.Movimiento1.Angulo2 = (KnotPoints.instance.theta2Array[0]);
296         cambios.Movimiento1.Angulo3 = (KnotPoints.instance.theta3Array[0]);
297
298         cambios.Movimiento2 = new Angulo();
299         cambios.Movimiento2.Angulo1 = (KnotPoints.instance.theta1Array[1]);
300         cambios.Movimiento2.Angulo2 = (KnotPoints.instance.theta2Array[1]);
301         cambios.Movimiento2.Angulo3 = (KnotPoints.instance.theta3Array[1]);
302
303         cambios.Movimiento3 = new Angulo();
304         cambios.Movimiento3.Angulo1 = (KnotPoints.instance.theta1Array[2]);
305         cambios.Movimiento3.Angulo2 = (KnotPoints.instance.theta2Array[2]);
306         cambios.Movimiento3.Angulo3 = (KnotPoints.instance.theta3Array[2]);
307
308         cambios.Movimiento4 = new Angulo();
309         cambios.Movimiento4.Angulo1 = (KnotPoints.instance.theta1Array[3]);
310         cambios.Movimiento4.Angulo2 = (KnotPoints.instance.theta2Array[3]);
311         cambios.Movimiento4.Angulo3 = (KnotPoints.instance.theta3Array[3]);
312
313         cambios.Movimiento5 = new Angulo();
314         cambios.Movimiento5.Angulo1 = (KnotPoints.instance.theta1Array[4]);
315         cambios.Movimiento5.Angulo2 = (KnotPoints.instance.theta2Array[4]);
316         cambios.Movimiento5.Angulo3 = (KnotPoints.instance.theta3Array[4]);
317
318         RequestBody requestBody = new RequestBody();
319         requestBody.collection = collection;
320         requestBody.id = id;
321         requestBody.changes = cambios;
322         StartCoroutine(PostRequest("http://34.206.7.229:3000/data", requestBody));
323
324     }
325
326     public void CallPostRequest2(string collection, string id, string field, string newValue) //Cambiar #
327     {
328         RequestBody2 requestBody = new RequestBody2();
329         requestBody.collection = collection;
330         requestBody.id = id;
331         requestBody.field = field;
332         requestBody.newValue = newValue;
333
```

```
334         StartCoroutine(PostRequest2("http://34.206.7.229:3000/control", requestBody));
335     }
336
337     public void CallGetRequest(string collection, string id)     //Leer Datos ingresando ID de intento
338     {
339
340         StartCoroutine(GetRequest("http://34.206.7.229:3000/data", collection, id));
341
342     }
343
344     public void CallGetRequest2(string collection, string id)   //Leer # de intentos
345     {
346         StartCoroutine(GetRequest2("http://34.206.7.229:3000/data", collection, id));
347
348     }
349
350     IEnumerator PostRequest(string url, RequestBody requestBody)
351     {
352         string json = JsonUtility.ToJson(requestBody);
353         Debug.Log(json);
354
355         // Convertir la cadena JSON en un byte array
356         byte[] bodyRaw = System.Text.Encoding.UTF8.GetBytes(json);
357
358         // Crear la solicitud POST con la URL y los datos JSON
359         using (UnityWebRequest www = UnityWebRequest.Put(url, bodyRaw))
360         {
361             www.method = UnityWebRequest.kHttpVerbPOST;
362             www.SetRequestHeader("Content-Type", "application/json");
363
364             yield return www.SendWebRequest();
365
366             if (www.result != UnityWebRequest.Result.Success)
367             {
368                 Debug.Log(www.error);
369             }
370             else
371             {
372                 Debug.Log(www.downloadHandler.text);
373             }
374         }
375     }
376
377     IEnumerator PostRequest2(string url, RequestBody2 requestBody)
378     {
379         string json = JsonUtility.ToJson(requestBody);
380
381         byte[] bodyRaw = System.Text.Encoding.UTF8.GetBytes(json);
```

```
382
383         using (UnityWebRequest www = UnityWebRequest.Put(url, bodyRaw))
384         {
385             www.method = UnityWebRequest.kHttpVerbPOST;
386             www.SetRequestHeader("Content-Type", "application/json");
387
388             yield return www.SendWebRequest();
389
390             if (www.result != UnityWebRequest.Result.Success)
391             {
392                 Debug.Log(www.error);
393             }
394             else
395             {
396                 Debug.Log(www.downloadHandler.text);
397             }
398         }
399     }
400
401     IEnumerator GetRequest(string url, string collectionName, string documentId)
402     {
403         string fullUrl = $"{url}?collection={collectionName}&id={documentId}";
404
405         using (UnityWebRequest www = UnityWebRequest.Get(fullUrl))
406         {
407             yield return www.SendWebRequest();
408
409             if (www.result != UnityWebRequest.Result.Success)
410             {
411                 Debug.Log(www.error);
412             }
413             else
414             {
415                 string json = www.downloadHandler.text;
416                 Movimiento responseData = JsonUtility.FromJson<Movimiento>(json);
417                 Debug.Log(responseData.Movimiento1.Angulo1);
418
419             }
420         }
421     }
422
423     IEnumerator GetRequest2(string url, string collectionName, string documentId)
424     {
425         string fullUrl = $"{url}?collection={collectionName}&id={documentId}";
426
427         using (UnityWebRequest www = UnityWebRequest.Get(fullUrl))
428         {
429             yield return www.SendWebRequest();
430
431             if (www.result != UnityWebRequest.Result.Success)
432             {
433                 Debug.Log(www.error);
434             }
435             else
436             {
437                 string json = www.downloadHandler.text;
438                 Intento responseData = JsonUtility.FromJson<Intento>(json);
439                 Debug.Log(responseData.Intentos);
440                 intento = int.Parse(responseData.Intentos);
441                 Debug.Log("Numero de Intento Actual Brazo 1: " + intento);
442                 inte.text = (4 - intento).ToString();
443
444             }
445         }
446     }
447 }
```

ANEXO E: TABLA DE RTT DE UNA PRUEBA PILOTO PARA LA OBTENCIÓN DE LA DESVIACIÓN ESTÁNDAR

Tiempo	Origen	Destino	RTT
0	192.168.3.96	34.206.7.229	97.390
0.1	192.168.3.96	34.206.7.229	97.854
0.2	192.168.3.96	34.206.7.229	98.315
2	192.168.3.96	34.206.7.229	98.440
2.1	192.168.3.96	34.206.7.229	98.396
9.5	192.168.3.96	34.206.7.229	104.412
9.6	192.168.3.96	34.206.7.229	100.520
9.7	192.168.3.96	34.206.7.229	97.497
9.8	192.168.3.96	34.206.7.229	96.904
9.9	192.168.3.96	34.206.7.229	98.031
10	192.168.3.96	34.206.7.229	96.758
10.1	192.168.3.96	34.206.7.229	97.105
10.2	192.168.3.96	34.206.7.229	96.930
10.3	192.168.3.96	34.206.7.229	96.996
10.5	192.168.3.96	34.206.7.229	96.817
10.6	192.168.3.96	34.206.7.229	96.697
10.7	192.168.3.96	34.206.7.229	97.106
10.8	192.168.3.96	34.206.7.229	97.058
10.9	192.168.3.96	34.206.7.229	96.762
11	192.168.3.96	34.206.7.229	96.920
11.1	192.168.3.96	34.206.7.229	97.057
11.2	192.168.3.96	34.206.7.229	96.944
16.8	192.168.3.96	34.206.7.229	96.796
16.9	192.168.3.96	34.206.7.229	96.661
17	192.168.3.96	34.206.7.229	97.006
18.4	192.168.3.96	34.206.7.229	96.576
18.5	192.168.3.96	34.206.7.229	97.961
18.6	192.168.3.96	34.206.7.229	96.880
19.4	192.168.3.96	34.206.7.229	97.189
19.5	192.168.3.96	34.206.7.229	97.025
19.6	192.168.3.96	34.206.7.229	96.974
20.7	192.168.3.96	34.206.7.229	96.519
20.8	192.168.3.96	34.206.7.229	97.146
20.9	192.168.3.96	34.206.7.229	97.781
21.1	192.168.3.96	34.206.7.229	97.771
21.2	192.168.3.96	34.206.7.229	97.535
21.3	192.168.3.96	34.206.7.229	97.066
21.5	192.168.3.96	34.206.7.229	97.752
21.6	192.168.3.96	34.206.7.229	97.555
21.6	192.168.3.96	34.206.7.229	97.100
22.8	192.168.3.96	34.206.7.229	97.615
22.9	192.168.3.96	34.206.7.229	98.797

23	192.168.3.96	34.206.7.229	96.958
23.1	192.168.3.96	34.206.7.229	96.832
23.2	192.168.3.96	34.206.7.229	96.699
23.3	192.168.3.96	34.206.7.229	97.219
23.4	192.168.3.96	34.206.7.229	96.386
23.5	192.168.3.96	34.206.7.229	96.894
23.7	192.168.3.96	34.206.7.229	97.400
23.8	192.168.3.96	34.206.7.229	97.065
23.9	192.168.3.96	34.206.7.229	96.920
24	192.168.3.96	34.206.7.229	97.685
26.5	192.168.3.96	34.206.7.229	97.203
26.5	192.168.3.96	34.206.7.229	97.399
26.6	192.168.3.96	34.206.7.229	97.190
27.7	192.168.3.96	34.206.7.229	97.353
27.8	192.168.3.96	34.206.7.229	97.160
28	192.168.3.96	34.206.7.229	96.710
28.1	192.168.3.96	34.206.7.229	96.947
28.2	192.168.3.96	34.206.7.229	96.961
28.7	192.168.3.96	34.206.7.229	96.504
28.8	192.168.3.96	34.206.7.229	97.841
28.9	192.168.3.96	34.206.7.229	97.071
29.5	192.168.3.96	34.206.7.229	96.677
29.7	192.168.3.96	34.206.7.229	97.317
29.8	192.168.3.96	34.206.7.229	97.804
29.9	192.168.3.96	34.206.7.229	96.666
30.4	192.168.3.96	34.206.7.229	96.327
30.5	192.168.3.96	34.206.7.229	96.863
30.6	192.168.3.96	34.206.7.229	97.050
30.7	192.168.3.96	34.206.7.229	97.082
30.9	192.168.3.96	34.206.7.229	96.781
31	192.168.3.96	34.206.7.229	97.018
31.1	192.168.3.96	34.206.7.229	96.977
31.2	192.168.3.96	34.206.7.229	96.350
31.3	192.168.3.96	34.206.7.229	97.872
31.4	192.168.3.96	34.206.7.229	97.021
35.4	192.168.3.96	34.206.7.229	97.221
38.8	192.168.3.96	34.206.7.229	97.078
38.8	192.168.3.96	34.206.7.229	97.683
38.9	192.168.3.96	34.206.7.229	96.950
39.6	192.168.3.96	34.206.7.229	104.150
39.7	192.168.3.96	34.206.7.229	97.017
39.7	192.168.3.96	34.206.7.229	97.897
39.8	192.168.3.96	34.206.7.229	97.024
39.9	192.168.3.96	34.206.7.229	96.742
40	192.168.3.96	34.206.7.229	96.990
40.1	192.168.3.96	34.206.7.229	96.820

40.2	192.168.3.96	34.206.7.229	98.537
40.3	192.168.3.96	34.206.7.229	97.582
40.4	192.168.3.96	34.206.7.229	96.851
41.7	192.168.3.96	34.206.7.229	96.754
41.8	192.168.3.96	34.206.7.229	96.959
41.9	192.168.3.96	34.206.7.229	96.822
43.6	192.168.3.96	34.206.7.229	97.071
46.9	192.168.3.96	34.206.7.229	96.417
46.9	192.168.3.96	34.206.7.229	98.272
47	192.168.3.96	34.206.7.229	96.866
48.3	192.168.3.96	34.206.7.229	102.460
48.4	192.168.3.96	34.206.7.229	97.099
48.5	192.168.3.96	34.206.7.229	96.816
48.6	192.168.3.96	34.206.7.229	96.728
48.8	192.168.3.96	34.206.7.229	97.205
48.9	192.168.3.96	34.206.7.229	96.904
49	192.168.3.96	34.206.7.229	97.146
49.1	192.168.3.96	34.206.7.229	96.850
49.2	192.168.3.96	34.206.7.229	97.344
50	192.168.3.96	34.206.7.229	97.404
50.1	192.168.3.96	34.206.7.229	97.712
50.2	192.168.3.96	34.206.7.229	96.887
50.3	192.168.3.96	34.206.7.229	97.166
50.5	192.168.3.96	34.206.7.229	97.159
50.6	192.168.3.96	34.206.7.229	96.885
50.6	192.168.3.96	34.206.7.229	97.948
50.7	192.168.3.96	34.206.7.229	96.893
50.9	192.168.3.96	34.206.7.229	97.079
51	192.168.3.96	34.206.7.229	97.641
51.2	192.168.3.96	34.206.7.229	97.593
51.5	192.168.3.96	34.206.7.229	96.557
51.6	192.168.3.96	34.206.7.229	96.943
51.7	192.168.3.96	34.206.7.229	96.858
51.8	192.168.3.96	34.206.7.229	96.894
51.9	192.168.3.96	34.206.7.229	97.041
52	192.168.3.96	34.206.7.229	96.924
52.1	192.168.3.96	34.206.7.229	96.980
52.2	192.168.3.96	34.206.7.229	96.870
58	192.168.3.96	34.206.7.229	97.932
Desviación estándar			1.1385

ANEXO F: TABLAS DE REGISTROS DE PARÁMETROS DE RENDIMIENTO

ESTUDIANTE MQTT					
Hora Del día	Máxima Latencia	Mínima Latencia	Latencia Promedio	Errores de Transmisión	Ancho de Banda
0:00	101	94	95.4	0	775
0:48	101	93	96.8	0	774
1:36	101	94	98	0	776
2:24	101	94	96.6	0	773
3:12	101	94	95.4	0	777
4:00	101	93	96.5	0	772
4:48	101	94	95.4	0	778
5:36	101	94	97.2	0	771
6:24	101	94	97.5	0	779
7:12	101	94	97.8	0	770
8:00	101	94	98.1	0	780
8:48	101	94	98.4	0	769
9:36	101	94	98.8	0	781
10:24	101	94	99.2	0	768
11:12	101	94	99.5	0	782
12:00	101	94	99.8	0	767
12:48	103	94	100.1	0	783
13:36	103	94	100.4	0	766
14:24	103	94	100.7	0	784
15:12	103	94	101	0	765
16:00	103	94	101.3	0	785
16:48	103	94	101.6	0	764
17:36	103	94	102.1	0	786
18:24	103	94	96.3	0	763
19:12	103	93	102.2	0	787
20:00	103	94	96.2	0	762
20:48	101	93	96.5	0	788
21:36	101	94	96.1	0	761
22:24	101	94	97.2	0	789
23:12	101	93	96	0	760

PROFESOR MQTT					
Hora Del día	máxima Latencia	mínima Latencia	Latencia Promedio	Errores de Transmisión	Ancho de Banda
0:00	140	94	100.4	0	2400
0:48	140	94	101.8	0	2399
1:36	140	94	101.1	0	2401
2:24	141	94	101.6	0	2398
3:12	139	94	100.4	0	2402
4:00	140	94	101.5	0	2397
4:48	140	94	100.4	0	2403
5:36	140	94	102.2	0	2396
6:24	140	94	102.5	0	2404
7:12	140	94	102.8	0	2395
8:00	140	94	103.1	0	2405
8:48	140	94	103.4	0	2394
9:36	140	94	103.8	0	2406
10:24	140	94	104.2	0	2393

11:12	140	94	104.5	0	2407
12:00	150	94	104.8	0	2392
12:48	160	96	105.1	0	2408
13:36	160	96	105.4	0	2391
14:24	160	98	105.7	0	2409
15:12	160	98	106	0	2390
16:00	160	94	106.3	0	2410
16:48	155	94	106.6	0	2389
17:36	160	94	107.1	0	2411
18:24	160	94	101.3	0	2388
19:12	165	94	107.2	0	2412
20:00	140	94	101.2	0	2387
20:48	140	94	101.5	0	2413
21:36	142	94	101.1	0	2386
22:24	140	94	102.2	0	2414
23:12	145	94	101	0	2385

ESTUDIANTE HTTP					
Hora Del día	máxima Latencia	mínima Latencia	Latencia Promedio	Errores de Transmisión	Ancho de Banda
0:00	120	98	99	2%	1711
0:48	120	98	96.8	2%	1710
1:36	120	98	100	2%	1712
2:24	121	98	99	2%	1709
3:12	120	98	99.5	2%	1713
4:00	122	98	99	2%	1708
4:48	120	98	99.5	2%	1714
5:36	120	98	99.75	3%	1707
6:24	120	98	99.8	3%	1715
7:12	120	98	100.2	3%	1706
8:00	120	98	100.5	3%	1716
8:48	120	98	100.8	3%	1705
9:36	125	98	101.1	2%	1717
10:24	126	98	101.4	3%	1704
11:12	130	98	101.7	3%	1718
12:00	125	98	102.1	2%	1703
12:48	135	98	102.4	3%	1719
13:36	125	98	102.7	3%	1702
14:24	128	98	103	3%	1720
15:12	130	98	103.3	3%	1701
16:00	120	98	103.6	3%	1721
16:48	118	98	103.9	3%	1700
17:36	120	98	104.2	3%	1722
18:24	120	98	104.5	3%	1699
19:12	120	98	102.2	3%	1723
20:00	122	98	99	2%	1698
20:48	120	98	99	2%	1724
21:36	121	98	99	2%	1697
22:24	121	98	99	2%	1725
23:12	122	98	99	2%	1700

Profesor HTTP					
Hora Del día	Máxima Latencia	Mínima Latencia	Latencia Promedio	Errores de Transmisión	Ancho de Banda

0:00	125	99	108	2%	3400
0:48	125	99	108	2%	3399
1:36	125	99	109	2%	3401
2:24	125	99	108	2%	3398
3:12	125	99	108	2%	3402
4:00	125	99	109	2%	3397
4:48	125	99	108	2%	3403
5:36	125	99	109	3%	3396
6:24	125	99	108	3%	3404
7:12	125	99	109	3%	3395
8:00	125	99	108	3%	3405
8:48	125	99	110	3%	3394
9:36	125	99	111	3%	3406
10:24	125	99	112	3%	3393
11:12	135	99	118	3%	3407
12:00	136	99	117	3%	3392
12:48	138	99	109	3%	3408
13:36	150	99	118	2%	3391
14:24	135	99	120	3%	3409
15:12	140	99	121	3%	3390
16:00	135	99	125	3%	3410
16:48	118	99	110	3%	3389
17:36	128	99	109	3%	3411
18:24	123	99	108	3%	3388
19:12	125	99	108	3%	3412
20:00	125	99	107	2%	3387
20:48	126	99	108	2%	3413
21:36	128	99	108	2%	3386
22:24	126	99	107	2%	3414
23:12	122	99	109	2%	3385

Estudiante				
	MQTT		HTTP	
Hora del día	% RAM promedio	% CPU promedio	% RAM promedio	% CPU promedio
0:00	45.713	2.874	51.418	2.851
0:48	45.589	2.727	51.404	2.996
1:36	45.498	2.890	51.432	2.979
2:24	45.678	2.754	51.422	2.888
3:12	45.756	2.829	51.418	2.942
4:00	45.502	2.917	51.421	2.846
4:48	45.635	2.776	51.429	2.907
5:36	45.718	2.733	51.486	2.879
6:24	45.547	2.787	51.421	2.932
7:12	45.622	2.810	51.401	2.904
8:00	45.491	2.761	51.417	2.938
8:48	45.555	2.849	51.410	2.922
9:36	45.687	2.734	51.396	2.874
10:24	45.756	2.897	51.395	2.936
11:12	45.624	2.712	51.412	2.932
12:00	45.472	2.788	51.415	2.949
12:48	45.712	2.791	51.445	2.903
13:36	45.523	2.878	51.469	2.869

14:24	45.789	2.743	51.475	2.931
15:12	45.667	2.792	51.416	2.913
16:00	45.489	2.854	51.463	2.898
16:48	45.578	2.803	51.426	2.905
17:36	45.695	2.776	51.415	2.946
18:24	45.632	2.868	51.438	2.891
19:12	45.539	2.707	51.423	2.937
20:00	45.690	2.814	51.415	2.922
20:48	45.578	2.726	51.446	2.906
21:36	45.485	2.865	51.517	2.893
22:24	45.635	2.825	51.496	2.962
23:12	45.721	2.809	51.463	2.893
PROMEDIO	45.619	2.803	51.433	2.915

Profesor				
	MQTT		HTTP	
Hora del día	% RAM promedio	% CPU promedio	% RAM promedio	% CPU promedio
0:00	50.064	3.318	52.340	3.382
0:48	50.124	3.289	52.507	3.468
1:36	50.018	3.345	52.577	3.390
2:24	50.212	3.256	52.257	3.495
3:12	50.096	3.324	52.390	3.458
4:00	50.043	3.312	52.512	3.451
4:48	50.185	3.278	52.620	3.411
5:36	50.076	3.299	52.335	3.314
6:24	50.113	3.312	52.504	3.380
7:12	50.049	3.278	52.610	3.426
8:00	50.155	3.301	52.476	3.410
8:48	50.134	3.333	52.283	3.421
9:36	50.199	3.245	52.307	3.436
10:24	50.043	3.300	52.594	3.438
11:12	50.087	3.323	52.592	3.416
12:00	50.166	3.267	52.360	3.414
12:48	50.010	3.312	52.600	3.379
13:36	50.232	3.322	52.290	3.446
14:24	50.057	3.289	52.318	3.358
15:12	50.142	3.334	52.475	3.413
16:00	50.121	3.233	52.330	3.493
16:48	50.054	3.345	52.600	3.344
17:36	50.178	3.355	52.287	3.420
18:24	50.065	3.311	52.411	3.481
19:12	50.199	3.245	52.490	3.405
20:00	50.090	3.299	52.294	3.419
20:48	50.202	3.311	52.462	3.378
21:36	50.116	3.344	52.536	3.307
22:24	50.043	3.288	52.297	3.446
23:12	50.159	3.299	52.410	3.396
PROMEDIO	50.114	3.302	52.435	3.413

ANEXO G: TABLAS DE REGISTROS DE PARÁMETROS DE RENDIMIENTO EN TIEMPO REAL

ESTUDIANTE MQTT tiempo real					
Hora Del día	Máxima Latencia	Mínima Latencia	Latencia Promedio	Errores de Transmisión	Ancho de Banda
0:00	143.5	98.5	99	1.6	7000
0:48	144.5	98.5	96	1.2	6999
1:36	143.5	98.5	95	1.1	7001
2:24	142.5	98.5	95	0.9	6998
3:12	143.5	98.5	95	1	7002
4:00	143.5	98.5	96	0.9	6997
4:48	143.5	98.5	96	1.1	7003
5:36	143.5	98.5	98	1.2	6996
6:24	143.5	98.5	99	1.4	7004
7:12	143.5	98.5	100	1.5	6995
8:00	143.5	98.5	99	1.3	7005
8:48	143.5	100.5	99	1.7	6994
9:36	148.5	99.5	101	1.8	7006
10:24	150.5	101.5	105	2	6993
11:12	154.5	100.5	103	1.9	7007
12:00	153.5	102.5	105	1.8	6992
12:48	158.5	101.5	108	2.1	7008
13:36	158.5	100.5	108	2	6991
14:24	166.5	100.5	107	2.2	7009
15:12	162.5	99.5	106	2.5	7000
16:00	165.5	102.5	106	2.3	7010
16:48	166.5	100.5	105	2.3	6989
17:36	161.5	101.5	103	2.5	7011
18:24	159.5	100.5	105	2.8	6988
19:12	148.5	99.5	106	2.7	7012
20:00	143.5	98.5	108	2.5	7000
20:48	143.5	98.5	108	2.3	7013
21:36	143.5	98.5	109	2.1	7000
22:24	144.5	98.5	107	2	7014
23:12	143.5	93.5	103	1.8	7001

ESTUDIANTE WEBSOCKET					
Hora Del día	Máxima Latencia	Mínima Latencia	Latencia Promedio	Errores de Transmisión	Ancho de Banda
0:00	145	99	108.4	3.2	5985
0:48	146	99	107	3.1	6014
1:36	145	99	108	3	5986
2:24	144	99	107.6	3.2	6013
3:12	145	99	107.1	3.3	5987
4:00	145	99	108	3.2	6012
4:48	145	99	108.4	3.1	5988
5:36	145	99	110.2	3.2	6011

6:24	145	99	110.5	3.3	5989
7:12	145	99	110.8	3.4	6010
8:00	145	99	111.1	3.5	5990
8:48	145	101	112	3.4	6009
9:36	150	100	111.8	3.5	5991
10:24	152	102	113	3.7	6008
11:12	156	101	112.5	3.6	5992
12:00	155	103	112.8	3.8	6007
12:48	160	102	113.1	4	5993
13:36	160	101	113.4	4.2	6006
14:24	168	101	112.5	4.1	5994
15:12	164	100	117	4	6005
16:00	167	103	114.3	4.2	5995
16:48	168	101	115	4.1	6004
17:36	163	102	115.1	4	5996
18:24	161	101	115.9	4.2	6003
19:12	150	100	115.2	3.8	5997
20:00	145	99	116	4	6002
20:48	145	99	116.3	3.9	5998
21:36	145	99	115.9	3.8	6001
22:24	146	99	113.4	3.9	5999
23:12	145	94	109	3.8	6000

PROFESOR MQTT					
Hora Del día	máxima Latencia	mínima Latencia	Latencia Promedio	Errores de Transmisión	Ancho de Banda
0:00	50	8	39.8	0.2	6000
0:48	50	8	39.5	0.2	5999
1:36	50	8	38.9	0.2	6001
2:24	50	8	39.1	0.1	5998
3:12	50	8	41.5	0.1	5992
4:00	50	8	42.3	0.1	5997
4:48	50	8	41	0.2	6003
5:36	50	8	41.5	0.3	5996
6:24	50	9	40.7	0.3	6004
7:12	50	8	42.7	0.4	5995
8:00	50	8	43.7	0.4	6005
8:48	50	8	44.8	0.5	5994
9:36	50	9	43.9	0.4	6006
10:24	50	8	48.7	0.4	5993
11:12	50	8	47.3	0.5	6007
12:00	50	9	46.2	0.5	5992
12:48	50	10	45.5	0.4	6008
13:36	50	8	44.8	0.5	5991
14:24	50	8	45.3	0.5	6009
15:12	50	8	44.6	0.5	5990
16:00	50	8	43.9	0.5	6010
16:48	50	9	43.2	0.4	6001
17:36	50	8	46.7	0.4	6011
18:24	50	10	45	0.3	5988
19:12	50	9	45	0.4	6022
20:00	50	8	46	0.4	6007
20:48	50	9	45	0.3	6023
21:36	50	8	44	0.4	5986
22:24	50	8	45	0.2	6014

23:12	50	7	40.5	0.2	5985

PROFESOR WEBSOCKET					
Hora Del día	máxima Latencia	mínima Latencia	Latencia Promedio	Errores de Transmisión	Ancho de Banda
0:00	53	38	43.5	0.2	6100
0:48	52	36	43.5	0.3	6099
1:36	53	37	42.5	0.2	6101
2:24	52	35	43.5	0.2	6098
3:12	54	38	44.5	0.3	6102
4:00	53	37	43.5	0.2	6097
4:48	51	38	43.5	0.2	6103
5:36	56	39	43.5	0.3	6096
6:24	54	38	43.5	0.4	6104
7:12	50	38	43.5	0.3	6095
8:00	52	39	43.5	0.4	6105
8:48	51	40	43.5	0.5	6094
9:36	50	41	43.5	0.6	6106
10:24	52	39	46.5	0.6	6093
11:12	51	38	45.5	0.6	6107
12:00	50	40	46.5	0.5	6092
12:48	51	41	47.5	0.5	6108
13:36	52	40	48.5	0.8	6091
14:24	53	41	47.5	0.6	6109
15:12	53	40	46.5	0.5	6090
16:00	54	41	45.5	0.5	6110
16:48	52	42	46.5	0.5	6089
17:36	50	40	45.5	0.5	6111
18:24	51	39	46.5	0.5	6088
19:12	52	38	43.5	0.5	6112
20:00	52	38	44.5	0.5	6087
20:48	51	37	43.5	0.5	6113
21:36	53	38	42.5	0.5	6086
22:24	51	37	43.5	0.5	6114
23:12	52	38	42.5	0.5	6085

Estudiante- Profesor				
Hora del día	MQTT		WEBSOCKET	
	% RAM promedio	% CPU promedio	% RAM promedio	% CPU promedio
0:00	56.248	4.945	58.253	5.015
0:48	56.169	5.021	58.144	5.081
1:36	56.117	4.968	58.285	5.054
2:24	56.275	4.934	58.197	5.080
3:12	56.18	5.005	58.251	5.006
4:00	56.232	4.961	58.129	5.074
4:48	56.116	5.032	58.283	5.113
5:36	56.283	4.913	58.219	5.049
6:24	56.132	4.985	58.135	5.166
7:12	56.226	5.009	58.301	5.031
8:00	56.243	4.927	58.152	5.097
8:48	56.159	4.97	58.260	5.060
9:36	56.206	5.016	58.136	5.124

10:24	56.128	4.943	58.308	5.027
11:12	56.269	4.99	58.170	5.159
12:00	56.291	4.952	58.213	5.058
12:48	56.212	5.038	58.307	5.021
13:36	56.119	4.923	58.153	5.108
14:24	56.301	4.995	58.250	5.037
15:12	56.25	5.041	58.130	5.140
16:00	56.31	4.93	58.241	5.174
16:48	56.249	5.02	58.295	5.025
17:36	56.322	4.944	58.146	5.105
18:24	56.168	5.027	58.236	5.023
19:12	56.235	4.978	58.148	5.110
20:00	56.281	4.952	58.296	5.024
20:48	56.183	4.907	58.213	5.123
21:36	56.195	4.994	58.260	5.009
22:24	56.281	5.033	58.218	5.147
23:12	56.138	5.002	58.297	5.058
PROMEDIO	56.217	4.979	58.221	5.077

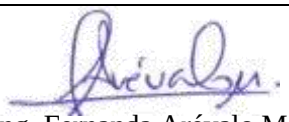
ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

DIRECCIÓN DE BIBLIOTECAS Y RECURSOS DEL APRENDIZAJE

UNIDAD DE PROCESOS TÉCNICOS

REVISIÓN DE NORMAS TÉCNICAS, RESUMEN Y BIBLIOGRAFÍA

Fecha de entrega: 08 / 01 / 2024

INFORMACIÓN DE LA AUTORA
Nombres – Apellidos: Cinthya Piedad Huilcamaygua De la Cruz
INFORMACIÓN INSTITUCIONAL
Facultad: Facultad de Informática y Electrónica
Carrera: Telecomunicaciones
Título a optar: Ingeniera en Telecomunicaciones
f. Analista de Biblioteca responsable:  Ing. Fernanda Arévalo M.