ISBN 978–987–633–574–4


# Análisis de Comunicaciones en Aplicaciones Móviles 3D para Domótica

Diego Encinas, Sebastián Dapoto, Federico Cristina, Cristian Iglesias, Federico Arias, Pablo Thomas, Patricia Pesado

Instituto de Investigación en Informática (III-LIDI). Facultad de Informática, Universidad Nacional de La Plata - Centro Asociado CIC. Buenos Aires, Argentina
{dencinas, sdapoto, fcristina}@lidi.info.unlp.edu.ar
{cristianniglesias, fede98.arias}@gmail.com
{pthomas, ppesado}@lidi.info.unlp.edu.ar

**Resumen** El presente trabajo expone el estudio y análisis de la performance en las comunicaciones de una aplicación móvil orientada a redes de sensores con tecnología en la Nube o *Cloud Computing*. La aplicación móvil reproduce un entorno visual 3D que está vinculado a diferentes dispositivos y sensores de una vivienda u oficina. Se desarrolló una comunicación bidireccional entre los dispositivos y sensores con la aplicación 3D desarrollada con la herramienta Unity. Se utilizaron servidores físicos como también virtuales (*Cloud Computing*). Además, se obtuvieron métricas de rendimiento de comunicaciones como latencia y throughput del sistema.

## 1 Introducción

Una vivienda domótica integra un conjunto de automatismos en cuanto a electricidad, electrónica, robótica, informática y telecomunicaciones, con el objetivo de asegurar al usuario una mejora en el confort, la seguridad, el ahorro energético, las facilidades de comunicación y las posibilidades de entretenimiento. La domótica se centra en dos puntos de vista: el del usuario y el tecnológico. Desde el punto de vista del usuario, una vivienda automatizada permite una mayor calidad de vida a través de las nuevas tecnologías, reduciendo el trabajo doméstico, mejorando el bienestar y, por ende, mejorando el control del consumo. Desde el punto de vista tecnológico, se encuentra la capacidad de los distintos objetos pertenecientes a una vivienda, con la posibilidad de intercomunicarse entre sí, a través de un soporte de comunicaciones [1]. El protocolo de comunicaciones más utilizado para la interconexión de los distintos dispositivos es el protocolo MQTT. Sus siglas en inglés corresponden a transporte de telemetría de cola de mensajes y es un protocolo abierto de máquina a máquina (M2M). Se caracteriza por ser orientado a mensajes permitiendo la comunicación entre los dispositivos de forma asincrónica y eficiente. Es uno de los estándares más utilizado para internet de las cosas (IoT, Internet of Things) [2]. En este trabajo, se explica el

ISBN 978-987-633-574-4




desarrollo de una comunicación bidireccional entre dispositivos móviles y sensores con una aplicación móvil 3D desarrollada mediante la herramienta Unity [3]. Además, se realiza un análisis por medio de métricas de rendimiento de comunicaciones del sistema.
Este trabajo se organiza del siguiente modo: a continuación se menciona la aplicación 3D generada; luego se plantean los componentes relacionados a comunicaciones del sistema. En el capítulo 4 se detalla el desarrollo de la comunicación bidireccional; seguido a esto se muestra la experimentación realizada. En el capítulo 6 se explican las métricas obtenidas. Finalmente se presentan los resultados y las conclusiones.

## 2 Aplicación móvil 3D

El cliente es una aplicación móvil 3D desarrollada en Unity. Éste es un framework de desarrollo de aplicaciones 3D que se destaca por la cantidad de documentación disponible, una numerosa y muy activa comunidad de usuarios, gran variedad de componentes pre-desarrollados (assets) y plugins que facilitan la integración con otras herramientas. Además, Unity ofrece variadas opciones de plataformas de publicación.
La aplicación móvil 3D permite recrear el conjunto de ambientes de una vivienda u oficina, incluyendo los objetos presentes en dichos espacios. Se cuenta con la posibilidad de manejar diferentes tipos de dispositivos, representados mediante hardware que simulan objetos del mundo real tales como ventiladores, luces, lámparas, televisores, entre otros. Entre las funciones con las que cuenta la aplicación se destacan:

- Creación de una casa. Al utilizar la aplicación por primera vez, es necesario crear la estructura de la casa que se desea controlar. La aplicación permite crear los modelos de todos los ambientes de la casa, con el nombre y tamaño correspondientes.
- Edición de una casa. En cualquier momento es posible modificar la estructura de una casa.
- Colocación de dispositivos. Mediante este módulo es posible seleccionar una posición en la pared, piso o techo de un ambiente determinado y colocar un dispositivo.
- Configuración. Permite entre otras cosas configurar la dirección del broker y asociar cada dispositivo con el sensor que lo maneja.
- Control de una casa. Este módulo permite visualizar los ambientes de una casa y controlar sus dispositivos presionando directamente sobre la pantalla. Como se puede observar en la ***Figura 1***, cuando se presiona sobre un electrodoméstico se accede a un menú con las posibles funciones del dispositivo.
- Conexión al Broker. Permite realizar la comunicación con el broker.

ISBN 978-987-633-574-4




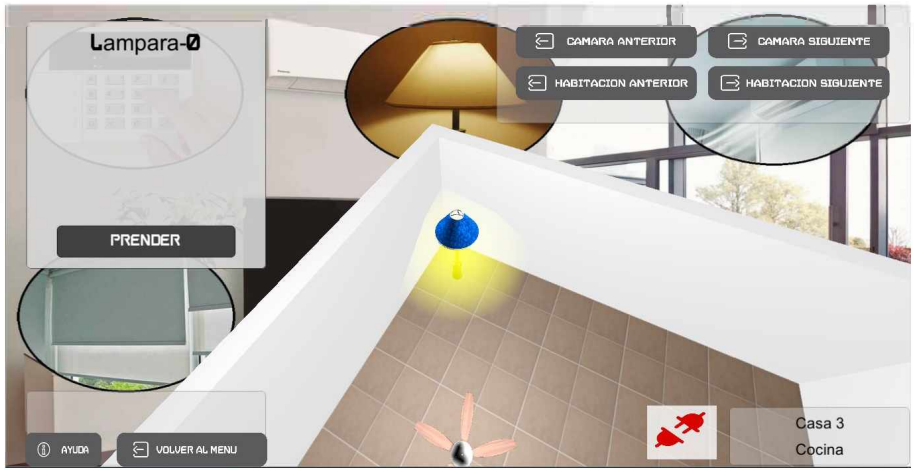


**Figura 1.** Control de dispositivos

## 3 Componentes del sistema

Para realizar la comunicación entre dispositivos y sensores, se cuenta con tres componentes principales:

- Componente de interfaz: es el encargado de la interacción con el usuario por medio de una aplicación móvil 3D mencionada en el capítulo 2.
- Componente de comunicación: es el responsable del procesamiento de las solicitudes y mensajes. Dicho procesamiento es realizado por un Broker de MQTT.
- Componente de control: permite llevar a cabo las solicitudes en el mundo físico que provengan del componente de interfaz. Los nodos (microcontroladores NodeMCU [4]) son adaptados a los diferentes dispositivos para proveer las señales necesarias y el funcionamiento deseado de cada uno.

### 3.1 Protocolo de comunicación

Para realizar la comunicación entre el cliente (desde la aplicación) y los componentes de control se utilizó un procesamiento de solicitudes y mensajes. Dicho procesamiento es realizado por un Broker de MQTT. MQTT es un protocolo de red liviano y simple del tipo publicación-suscripción, permite ser utilizado por dispositivos de recursos limitados y que no dispongan de gran ancho de banda. Este protocolo se monta típicamente sobre TCP/IP [5]. MQTT define dos entidades de red, un servidor, llamado Broker, y un número de clientes conectados a dicho Broker. Los mensajes se organizan por tópicos y funcionan de la siguiente manera: cuando un cliente quiere realizar una publicación, debe definir el tópico al cual será publicada, el Broker se encargará de distribuir el mensaje enviado entre todos los clientes que se hayan suscripto a dicho tópico. Un tópico puede

ISBN 978-987-633-574-4




contener varios subtópicos utilizando "/", por ejemplo "Habitación0/Lámpara-1". Los datos enviados por la aplicación contienen la información necesaria para que el nodo receptor analice y realice la acción correspondiente para modificar los dispositivos físicos. El tópico estará compuesto por RESIDENCIA/HABITACIÓN/DISPOSITIVO, por ejemplo, si se tiene una residencia llamada Juan con una habitación y se quiere realizar el apagado/encendido de una luz, el tópico generado será Juan/Habitación-0/Luz-1. A su vez se tendrá otro tópico el cual se usará para que el microcontrolador (nodo) envíe el estado del dispositivo, permitiendo saber si se ha manipulado de manera externa a la aplicación. Se tendrán, entonces, dos tópicos, uno que envía información de la aplicación al nodo de control, y otro que envía información del estado de un determinado dispositivo, del nodo de control a la aplicación.

## 4 Comunicación Bidireccional

Bajo la comunicacion bidireccional definida, tanto el cliente como los nodos de control envıan y reciben mensajes. Estos mensajes son almacenados en un servidor el cual los distribuirá a través de la red. La posibilidad de enviar y recibir mensajes permite un mejor control del estado de los dispositivos. Es decir, no solo se tiene el control interno desde la aplicación, sino que también se cuenta con la información de las alteraciones realizadas a los objetos físicos de forma externa, y por lo tanto, es posible realizar las acciones necesarias para que el estado del objeto se visualice de forma correcta en tiempo real.
Los firmwares de los NodeMCU y su depuración fueron realizados por medio de la herramienta Arduino IDE. Este entorno multiplataforma fue desarrollado en java, y se lo publica bajo la Licencia Pública General de GNU, admite lenguajes C y C++, y adicionalmente cuenta con una biblioteca de software, que proporciona un conjunto de procedimientos de E/S [6]. Además, se utilizó un servidor en la nube, Amazon Cloud [7], en el cual se instaló Mosquitto, un agente de mensajes de código abierto que implementa el protocolo MQTT [8].
En la ***Figura 2*** se muestra la bidireccionalidad adquirida, la aplicación publicará un mensaje según el accionar del usuario (Unity), mientras que desde el nodo de control (desarrollado en Arduino) se publicará el estado en el que se encuentra el dispositivo físico. Estos mensajes son distribuidos desde un servidor en la nube (AWS) o en un servidor local.

A cada dispositivo físico le corresponde un nodo de control, es decir, cada nodo controla un único objeto, por lo tanto, cada uno tiene un tópico que lo identifica. Al iniciar la aplicación, se envían al Broker los tópicos pertenecientes a cada objeto. Cada nodo identifica y se suscribe al tópico que le corresponde. Además, para poder recibir el estado de los dispositivos físicos en tiempo real, la aplicación también se suscribe a los tópicos en los cuales los nodos publicarán el estado de cada dispositivo. Esta última funcionalidad permite que la aplicación esté notificada en todo momento sobre lo que sucede en el mundo real, y en base a ello pueda realizar las modificaciones necesarias dentro de la aplicación, brindando información actualizada al usuario.

ISBN 978–987–633–574–4




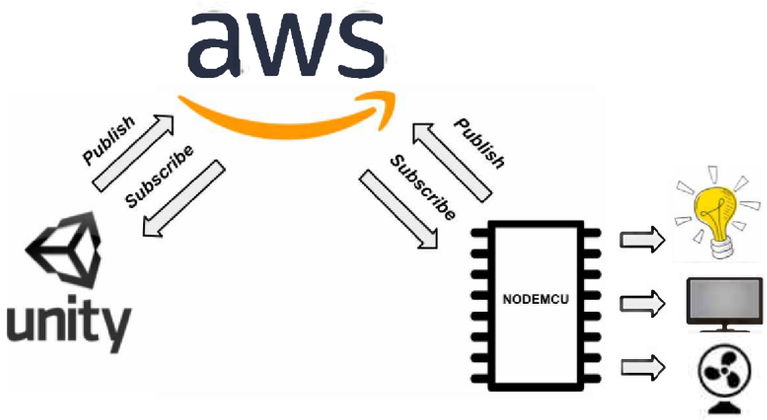


**Figura 2.** Bidireccionalidad

## 5 Experimentación

Una aplicación con estas características permite ser utilizada en cualquier ambiente, ya sea un hogar, un edificio, una oficina. Además, la funcionalidad de comunicación de la aplicación puede estar ligada a dos tipos de servidores, local o en la nube, por lo que es posible analizar el comportamiento de cada tipo de servidor en un mismo escenario de prueba. Esto permite tener una visión de cómo se comporta el protocolo de comunicación utilizado, como también la logística de programación a medida que se lo fuerza con solicitudes de mensajes. Como servidor local se utilizó una notebook con sistema operativo Ubuntu, y para el servidor en la nube se utilizó la plataforma Amazon Web Service, que proporciona Amazon EC2, servicio web que brinda capacidad de procesamiento en la nube [9].

Como funcionalidad básica de prueba se optó por realizar el encendido/apagado de una luz. Como se puede observar en la ***Figura 3***, el mensaje inicia en el Cliente (Tablet), se envía al servidor o broker (AWS), y desde alli al Cliente (NodeMCU). Este último responde de dos formas: por un lado envía un mensaje de vuelta indicando el estado de la luz y por otro lado enciende/apaga dicha luz. Aprovechando la comunicación bidireccional, se espera que la tasa de recepción de mensajes por parte del nodo de control y la aplicación, sea la misma que la tasa a la cual se envían los mensajes.

Los mensajes estaban compuestos por dos palabras alternadas, “ON” correspondientes a 2 bytes y “OFF” correspondientes a 3 bytes. Las pruebas tienen un total de envío de 100 mensajes. Se enviaron diferentes cantidades de mensajes en cada segundo. El escenario de prueba se basó en el envío de 1, 2, 4, 5 y 10 mensajes independientemente, cada 1 segundo. Estos mensajes se enviaban al Broker

ISBN 978–987–633–574–4




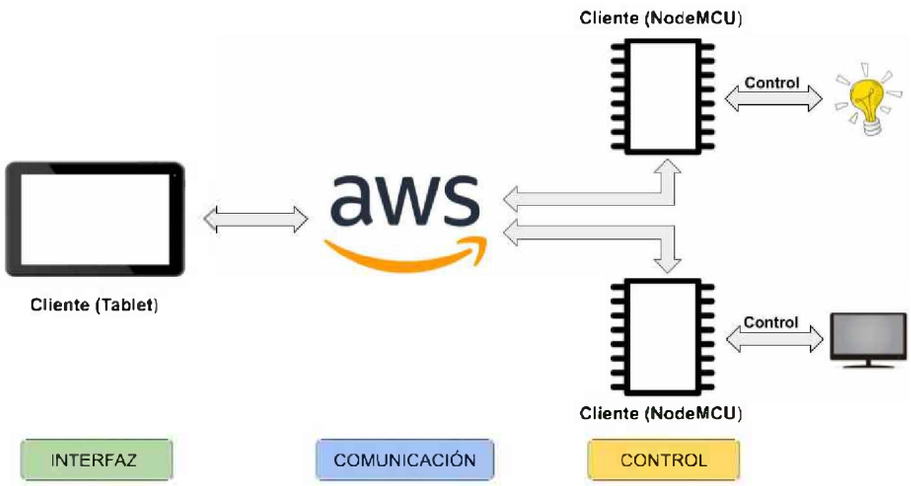


**Figura 3.** Relación entre los componentes

mediante el tópico correspondiente desde el cliente, es decir, la aplicación móvil. Los mensajes se recibian por el NodeMCU en el cual se registraba el tiempo de llegada de cada mensaje a partir del último recibido. El microcontrolador, analiza el mensaje arribado y responde con otra publicación en un nuevo tópico. La aplicación está suscripta a este tópico, por lo que se procede a tomar el tiempo en el que llega el mensaje del lado del cliente. Asi, se tiene una aproximación del tiempo de envío, como también, del tiempo de respuesta al mensaje arribado al microcontrolador. A partir de la suma de estos dos valores, en milisegundos, es posible obtener el tiempo que tarda el mensaje en llegar y la respuesta del mismo dando como resultado la latencia de la comunicación bidireccional. Con los datos medidos se generaron métricas de rendimiento de comunicación como latencia, throughput, entre otros.

- La latencia se entiende como la suma de tiempos entre el momento en el que un mensaje es publicado y el momento en el que dicho mensaje es recibido por el suscriptor y vuelto a publicar llegando al suscriptor quien es el que había iniciado los envíos de mensajes.
- El throughput se entiende como la tasa de paquetes que se envían a través de un canal de comunicación.

## 6 Métricas obtenidas

Para una mejor comprensión del tipo de experimentación realizada, a continuación se detallan especificamente los mecanismos de resolución y análisis utilizados en las diferentes etapas de las dos primeras pruebas. Las métricas resultantes se reflejan en graficas que muestran el comportamiento del canal de comunicación

ISBN 978-987-633-574-4




del sistema, observándose los tiempos de arribo de los mensajes a la aplicación móvil 3D y al nodo de control, como también, el número del mensaje enviado.

### 6.1 Servidor en la nube

Para llevar a cabo las pruebas en un servidor en la nube, se utilizó la plataforma Amazon Web Services con la región ubicada en América del Sur (São Paulo) sa-este-1. Para confirmar la correcta comunicación con el servidor, se realizó un ping desde la dirección IP con la que se realizaron todas las pruebas. El tiempo de respuesta fue de 33 milisegundos y ningún paquete perdido.

**1ra Prueba:**

Se comenzó con el envío de 1 mensaje cada 1 segundo, con un total de 100 mensajes. Por lo tanto 50 mensajes fueron de 2 bytes y los 50 restantes de 3 bytes, teniendo un envío de 2000 bits en total. El tiempo promedio entre mensajes recibidos en el nodeMCU fue de 1019 milisegundos, en relación con el tiempo promedio en la aplicación que fue de 962 milisegundos. Por lo tanto se tiene una latencia en la bidireccionalidad de 1,981 segundos. Esto indica que los envíos se hicieron de forma normal, ya que como se envían mensajes cada 1 segundo al nodo de control, este recibe cada mensaje cada 1 segundo, y por lo tanto responderá también cada 1 segundo, logrando un tiempo total de aproximadamente de 2 segundos en el envío y respuesta.

En la ***Figura 4*** se puede observar la relación de los intervalos de tiempo entre mensajes en el microcontrolador (Color azul) y la aplicación (Color naranja).

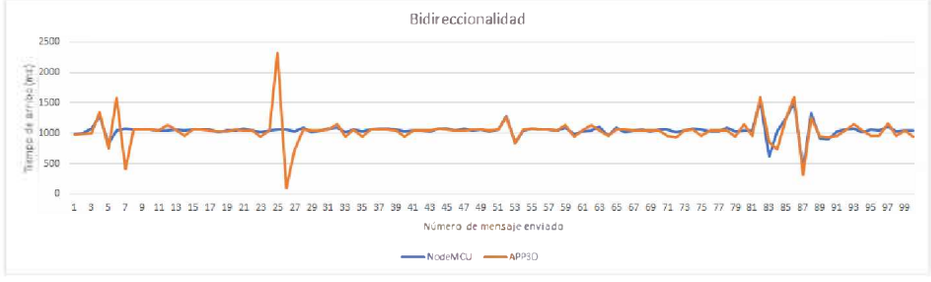


**Figura 4.** Intervalo de tiempos entre mensaje y mensaje (1ra prueba)

En esta prueba no se observó que se hayan perdido mensajes en ningún momento. Se puede apreciar que hay momentos que del lado de la aplicación el tiempo de arribo de mensajes fue superior a lo normal. Sin embargo, una vez recibido el mensaje, el siguiente mensaje llega al instante. Puede existir una demora en el tiempo de procesamiento o análisis de los mensajes cuando éstos arriban al microcontrolador, provocando que los próximos mensajes a arribar se acumulen en la cola del buffer del microcontrolador. Una vez finalizado el análisis y vuelto a publicar, es posible que los mensajes que se encuentran en el buffer hayan sido analizados más rapidamente, haciendo que su envío sea

ISBN 978–987–633–574–4




casi instantáneo. Lo explicado anteriormente se puede visualizar en la ***Figura 4*** en los intervalos entre los mensajes 24,25 y 25,26. Se destaca que gráficamente, tanto el envío desde la aplicación y la respuesta desde el microcontrolador siguen aproximadamente la misma tasa de respuesta.

**2da Prueba:**

Se enviaron 2 mensajes cada 1 segundo con un total de 100 mensajes. Del lado del nodo (nodeMCU) se recibieron 100 mensajes, 50 de 2 bytes y el resto de 3 bytes. Mientras que del lado de la aplicación se recibieron 99 mensajes dando por sentado que se perdió 1 mensaje. Cómo se enviaron 2 mensajes por segundo, el tiempo entre mensaje y mensaje fue de 0.5 segundos.

Si se reciben 2 mensajes cada 1 segundo, y se tiene un total de 100 mensajes, el tiempo que debería transcurrir en recibir dichos mensajes es de 50 segundos. En las pruebas se tiene un total de 52 segundos aproximadamente de ambos lados en recibir los mensajes, teniendo un retraso de casi 2 segundos. El tiempo promedio de recepción de mensaje en el nodo fue de 516 ms, mientras que en la aplicación fue de 510 ms.

En la ***Figura 5*** se pueden observar picos que corresponden a lo explicado en la primera prueba.

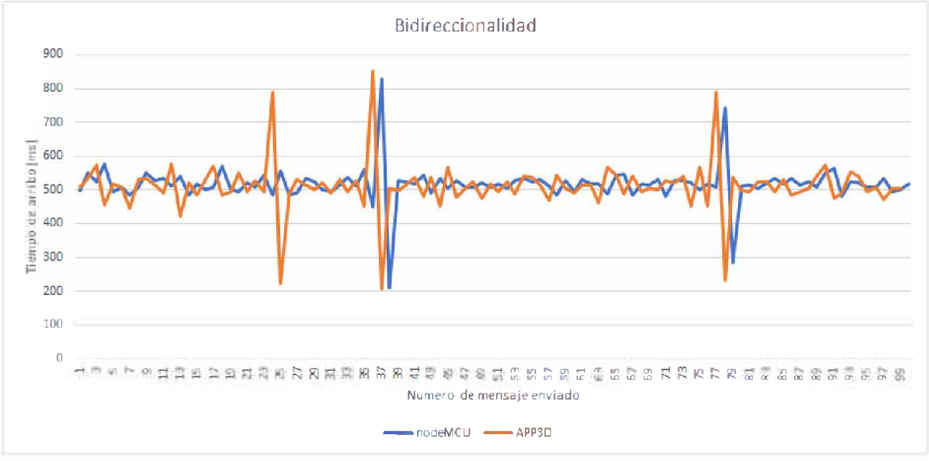


**Figura 5.** Intervalo de tiempo entre mensajes (2da prueba)

Finalmente, teniendo en cuenta los datos obtenidos se destaca que no se tuvo exigencia en la funcionalidad de la comunicación, pero se perdió un mensaje en el último envió por parte de la aplicación.

### 6.2 Servidor local

Como se mencionó en el apartado Experimentación, como servidor local se utilizó una notebook con sistema operativo Ubuntu. Dicha notebook se encontraba bajo la misma red local que el microcontrolador nodeMCU o nodo de control.

ISBN 978–987–633–574–4




**1er Prueba**

La prueba es similar a la realizada en la sección anterior en donde se envía 1 mensaje cada 1 segundo. El tiempo de recepción de mensajes del lado del nodeMCU fue de 1012 ms y del lado de la aplicación de 1005 ms. El tiempo total que llevó el envío de mensajes de la aplicación al nodo fue de 101 segundos. En la ***Figura 6*** se muestran las respuestas de arribo de mensajes.


**Figura 6.** Intervalo de tiempo entre mensaje y mensaje (1ra prueba)

Se puede apreciar claramente que el comportamiento bidireccional de la comunicación es casi exactamente igual, con aproximadamente el mismo tiempo de respuesta, arribaron los 100 mensajes y no se perdió ninguno. El tiempo de arribo en ambos lados es casi una constante.

**2da Prueba**

Se enviaron 2 mensajes por segundo, es decir, 1 mensaje cada 500 milisegundos. El tiempo promedio de arribo del lado del nodeMCU fue de 525 milisegundos y del lado de la aplicación de 522 milisegundos. En la ***Figura 7*** se puede observar que el tiempo de arribo en ambos lados fue similar en gran parte de la comunicación. Solo se generaron picos de retardo en ciertos instantes, pero no afectaron a la comunicación bidireccional.

## 7 Resultados

En el ***Cuadro 1*** se resumen los resultados obtenidos en las 5 pruebas realizadas utilizando el servidor en la nube. En cuanto a las pruebas utilizando un servidor local, en el ***Cuadro 2*** se resumen los 6 resultados obtenidos.

ISBN 978–987–633–574–4




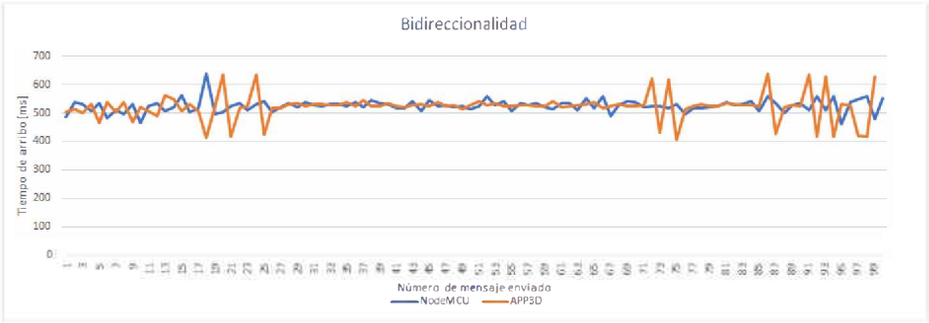


**Figura 7.** Intervalo de tiempo entre mensaje y mensaje (2da prueba)

**Cuadro 1.** Resultados utilizando un servidor en la nube.

| Número de Prueba | Throughput | Tiempo Promedio Bidireccional [ms] | Número de mensajes perdidos enviados a la aplicación |
|---|---|---|---|
| 1 | 1 msj/seg | 2093 | 0 |
| 2 | 2 msj/seg | 1028 | 1 |
| 3 | 4 msj/seg | 595 | 2 |
| 4 | 5 msj/seg | 451 | 10 |
| 5 | 10 msj/seg | 284 | 25 |

**Cuadro 2.** Resultados utilizando un servidor local.

| Número de Prueba | Throughput | Tiempo Promedio Bidireccional [ms] | Número de mensajes perdidos enviados a la aplicación |
|---|---|---|---|
| 1 | 1 msj/seg | 2017 | 0 |
| 2 | 2 msj/seg | 1047 | 0 |
| 3 | 4 msj/seg | 526 | 1 |
| 4 | 5 msj/seg | 420 | 1 |
| 5 | 10 msj/seg | 232 | 1 |
| 6 | 1 msj/0,05 seg | 159 | 23 |

Estas pruebas surgieron de la necesidad de analizar el rendimiento de la comunicación bidireccional desarrollada. Por lo tanto, se procuró obtener métricas de comunicaciones entre la aplicación móvil 3D y un nodo de control. Como conclusión, se destaca que, al ser una comunicación bidireccional los tiempos de envío y recepción de mensajes tuvieron la misma tasa de respuesta, que era lo esperado. La utilización de dos servidores posibilitó ver que tan confiable es establecer una comunicación con una determinada cantidad de mensajes por segundo. Se destaca que un servidor local tiene una muy buena respuesta en todo

ISBN 978-987-633-574-4




tipo de escenarios, comparado con un servidor en la nube, en donde en ciertas pruebas hubo perdida de mensajes. Adicionalmente, considerando que el servidor utilizado se encuentra en Brasil y las pruebas fueron realizadas desde la ciudad de Berisso (Argentina), es posible que los tiempos de respuesta disminuyan en caso de utilizar servidores con mayor cercanía.
Finalmente, la aplicación desarrollada es apta para ser implementada en cualquier entorno. Sin embargo, teniendo en cuenta el análisis realizado, se podría decir que un servidor local sería la elección correcta en el caso de una vivienda. Aunque, el usuario debería contar con conocimientos específicos ante la aparición de algún inconveniente en el servidor. Otra observación es que debido a la situación sanitaria actual (COVID-19) los precios de los servicios en la nube han disminuido considerablemente. Entonces, con la elección de un servidor en la nube se podría desligar al usuario del mantenimiento requerido [10].
Por otro lado, existe la posibilidad de implementar el sistema en un edificio lo que implica la utilización de una gran cantidad de nodos (NodeMCU). Considerando la capacidad de procesamiento que se obtuvo en las pruebas, la utilización de un servidor en la nube podría llevar a la saturación del canal de comunicaciones y la consecuente perdida de paquetes. Para este tipo de instalaciones, un servidor local permitiría una menor exigencia al mismo, una comunicación más fluida y un menor tiempo de respuesta.

## 8 Conclusiones

En el presente trabajo se ha realizado un análisis e investigación relacionada especificamente a las comunicaciones de un sistema de control domótico a través de aplicaciones móviles.
Se han desarrollado e implementado diversos módulos que corresponden a la aplicación cliente, servidor en la nube y sensores (NodeMCU). Este aporte ha permitido realizar el control del estado de los diferentes dispositivos electrónicos (nodos de control) en tiempo real y visualizarlo en un modelo tridimensional en la aplicación móvil, utilizando una arquitectura en la nube como servidor.
A partir de las pruebas experimentales, con servidores físicos y virtuales, se han obtenido métricas para cuantificar el rendimiento del flujo de mensajes, permitiendo validar y seleccionar posibles configuraciones del sistema.
Como trabajo futuro se propone realizar un análisis del rendimiento de las comunicaciones utilizando otras infraestructuras en la Nube existentes, especificamente orientadas a Edge Computing y 3D.

## Referencias

1. Mateus Cruz, Diego Alejandro. Molina Castañeda, Jonathan Esteban. Jaramillo Benavides, Maria Camila, “Domótica, el hogar digital,” 2018. [Online]. Available: https://answersingenesis.org/answers/magazine.Accedido:4-2021
2. MQTT, “Scalagent. benchmark of mqtt servers.” 2015. [Online]. Available: https://mqtt.org/.Accedido:4-2021

ISBN 978–987–633–574–4




3. Unity, "Unity homepage." [Online]. Available: https://unity.com/.Accedido:4-2021
4. NodeMCU, "Datasheet nodemcu." [Online]. Available: https://www.esploradores.com/datasheet-nodemcu/.Accedido:4-2021
5. R. A. Light, "Mosquitto: server and client implementation of the mqtt protocol," *Journal of Open Source Software, 2(13), 265*, 2017.
6. N. Zlatanov, "Arduino and open source computer hardware and software," *IEEE Computer Society*, 2005.
7. AWS, "Página oficial amazon web services." [Online]. Available: https://aws.amazon.com/es/.Accedido:4-2021
8. MQTT, "Mqtt homepage." [Online]. Available: https://mosquitto.org/.Accedido:4-2021
9. AWS, "Amazon ec2. capacidad informática segura y de tamaño ajustable que admite prácticamente cualquier carga de trabajo." [Online]. Available: https://aws.amazon.com/es/ec2/?ec2-whats-new.sort-by=item.additionalFields.postDateTime&ec2-whats-new.sort-order=desc.Accedido:4-2021
10. Telesemana, "Telesemana.com. la pandemia impulsa el consumo de la nube en el tercer trimestre de este año," Diciembre 2020. [Online]. Available: https://www.telesemana.com/blog/2020/10/30/la-pandemia-impulsa-el-consumo-de-la-nube-en-el-tercer-trimestre-de-este-ano/.Accedido:4-2021