ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

REDES DE SENSORES INALÁMBRICOS PARA IOT

IMPLEMENTACIÓN DEL DIAGRAMA DE ESTADOS DEL CONVERTIDOR DE PROTOCOLOS MQTT-SN A MQTT UTILIZANDO UML

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN TELECOMUNICACIONES

CARLOS ALBERTO PANCHI CORNEJO

carlos.panchi@epn.edu.ec

DIRECTOR: M.SC. CARLOS ROBERTO EGAS ACOSTA

carlos.egas@epn.edu.ec

DMQ, Enero 2022

CERTIFICACIONES

Yo, Carlos Alberto Panchi Cornejo declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

CARLOS ALBERTO PANCHI CORNEJO

Certifico que el presente trabajo de integración curricular fue desarrollado por Carlos Alberto Panchi Cornejo, bajo mi supervisión.

CARLOS ROBERTO EGAS ACOSTA
DIRECTOR

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el producto resultante del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

CARLOS ALBERTO PANCHI CORNEJO

CARLOS ROBERTO EGAS ACOSTA

DEDICATORIA

A Dios, mis padres, hermanos y a mi novia.

AGRADECIMIENTO

Agradezco a Dios y a la Virgen Dolorosa por cuidar mis pasos y sobre todo por brindarme la dicha de tener a mis padres junto a mí. A pesar de todas las adversidades ambos tienen salud y siguen acompañándome.

A mis padres, Segundo y Janeth, por ser mi más grande apoyo durante toda mi vida, por apoyarme en mi desarrollo académico e inculcarme valores para mi crecimiento personal. Gracias a ellos he logrado cumplir muchas metas, todo logro que alcance siempre se los dedicare en primer lugar. A mi hermana Joselyn y mis hermanos Billy y Javier, por estar a mi lado en los buenos y malos momentos.

A mi novia, Digna Isabel, por ser el motivo de mi felicidad y mi más grande apoyo en todo aspecto. Sin su amor y paciencia no lo hubiese logrado, mi mayor deseo es cumplir todas nuestras metas y sueños juntos.

Quiero expresar mi agradecimiento a mi director, Carlos Egas, por su tiempo y su colaboración en el desarrollo de este trabajo.

ÍNDICE DE CONTENIDO

CERTIFICACIONES I
DECLARACIÓN DE AUTORÍA II
DEDICATORIA III
AGRADECIMIENTO IV
ÍNDICE DE CONTENIDO V
RESUMEN VII
ABSTRACT VIII
1 INTRODUCCIÓN 1
1.1 Objetivo general 2
1.2 Objetivos específicos 2
1.3 Alcance 2
1.4 Marco teórico 4
1.4.1 Redes inalámbricas 4
1.4.2 Estándar 802.15.4 4
1.4.3 Redes inalámbricas de sensores 4
1.4.4 Internet de las cosas (IoT) 5
1.4.4.1 Protocolos para IoT 6
1.4.4.1.1 Protocolo MQTT 6
1.4.4.1.2 Protocolo MQTT-SN 10
1.4.5 Lenguaje Unificado de Modelado (UML) 14
1.4.5.1 Diagramas de Estado 14
1.4.6 UPPAAL Model Checker 15
2 METODOLOGÍA 16
2.1 Conversión de protocolo MQTT-SN a MQTT 16
2.2 Conversión de protocolo MQTT a MQTT-SN 35
2.3 Desarrollo de los diagramas de estado del convertidor de protocolos en UPPAAL Model Checker 45
2.3.1 Convertidor de protocolo MQTT-SN a MQTT 45
2.3.2 Convertidor de protocolo MQTT a MQTT-SN 46
3 RESULTADOS 48
3.1 Validación del comportamiento del diagrama de estados del convertidor de protocolo MQTT-SN a MQTT 48
3.2 Validación del comportamiento del diagrama de estados del convertidor de protocolo MQTT a MQTT-SN 53

4 CONCLUSIONES Y RECOMENDACIONES 55
4.1 Conclusiones 55
4.2 Recomendaciones 56
5 REFERENCIAS BIBLIOGRÁFICAS 57
6 ANEXOS 59
ANEXO I 60

RESUMEN

La finalidad de este trabajo de integración curricular es implementar los diagramas de estados del convertidor de protocolos MQTT-SN a MQTT y viceversa utilizando el lenguaje de modelado unificado (UML).

Para la implementación de los diagramas, se crean estados que contienen procesos asociados al funcionamiento del convertidor de protocolos en cada caso. Previo a la definición de los estados se realiza una revisión teórica y funcional de los protocolos MQTT y MQTT-SN para brindar una comprensión total de los mensajes incluidos en cada uno, así como de los campos de mensaje que interactúan en el proceso de conversión.

Los diagramas de estados obtenidos como resultado de este trabajo están diseñados suponiendo el caso de un arribo de mensaje desde un componente de red que maneja un protocolo y requiere comunicarse con otro que maneja un protocolo diferente. El evento de llegada de mensajes activa los cambios de estados y desencadena una seria de procesos que concluyen en la generación de un mensaje con el protocolo adecuado para el destino.

Posterior a la implementación de los diagramas de estados, con la intención de verificar el resultado de este trabajo de integración curricular para crear una referencia para futuros proyectos, se realizan pruebas de funcionamiento utilizando la herramienta UPPAAL para probar el comportamiento de cada diagrama y así encontrar inconsistencias o solucionar problemas de lógica en los estados y procesos definidos.

PALABRAS CLAVE: Diagrama de estados, Convertidor de protocolos, MQTT, MQTT-SN, UML, UPPAAL.

ABSTRACT

The purpose of this curricular integration work is to implement the state diagrams of the protocol converter from MQTT-SN to MQTT and vice versa using the unified modeling language (UML).

States are created for the implementation of the diagrams that contain processes associated with the operation of the protocol converter in each case. Prior to the definition of the states, a theoretical and functional review of the MQTT and MQTT-SN protocols is carried out to provide a complete understanding of the messages included in each one, as well as of the message fields that interact in the conversion process.

The state diagrams developed in this work are designed to assume the case of a message arrival from a network component that handles a specific protocol and requires communication with another that handles a different protocol. The message arrival event activates the state changes and triggers a series of processes that conclude in the generation of a message with the appropriate protocol for the destination.

After the implementation of the state diagrams, with the intention of verifying the result of this curricular integration work to create a reference for future projects, performance tests are performed using the UPPAAL tool to test the behavior of each diagram in order to find inconsistencies or solve logic problems in the defined states and processes.

KEYWORDS: State diagram, Protocol converter, MQTT, MQTT-SN, UML, UPPAAL.

1 INTRODUCCIÓN

El constante desarrollo tecnológico ha influenciado en la innovación dentro de varias ramas de la ingeniería, específicamente en las telecomunicaciones, debido a que esta área es imprescindible para las personas y se encuentra en continua evolución buscando una comunicación rápida y eficiente entre usuarios.

Dentro de las telecomunicaciones se debe hacer énfasis en las comunicaciones inalámbricas ya que estas han permitido establecer una comunicación efectiva entre el humano y su entorno, permitiendo la transmisión de información proveniente de la medición y monitoreo de los fenómenos físicos que nos rodean.

Las comunicaciones inalámbricas han propiciado el desarrollo de las redes inalámbricas de sensores (WSN – Wireless Sensor Networks), las cuales son masivamente utilizadas en la actualidad y su crecimiento es exponencial gracias a la gran variedad de aplicaciones que posee.

Entre las aplicaciones de las comunicaciones inalámbricas en el área de sensores se encuentra el internet de las cosas (IoT – Internet of Things) el cual abre la posibilidad de un nuevo “entorno conectado”, donde todos los objetos se puedan comunicar y puedan brindar facilidades de automatización y monitoreo a los usuarios. Esto ha impulsado el surgimiento de nuevas aplicaciones que requieren redes inalámbricas de sensores, conformadas por dispositivos de recursos limitados tanto de batería como procesamiento, pero con la capacidad de utilizar las nuevas arquitecturas y protocolos desarrollados en IoT [1].

Las aplicaciones que tienen los requerimientos antes mencionados necesitan que la información que circula en la red se entregue a los receptores según un tema de interés y no según una dirección en la red, esto se conoce como una comunicación centrado en los datos y encaja con las limitaciones de procesamiento en los dispositivos de red [2].

Una comunicación centrada en los datos es el modelo Publicación/Suscripción o también conocida como “Pub/Sub”, conocida como un sistema de mensajería ampliamente usado y escalable. En IoT existen los protocolos MQTT (Message Queue Telemetry Transport) [3] y una extensión creada para redes de sensores llamada MQTT-SN (Message Queue Telemetry Transport – Sensor Networks) [2]. Ambos son protocolos “pub/sub” y pueden ser usados en una misma aplicación, pero requieren un proceso de conversión de protocolos que se debe realizar en la red para poder garantizar su interoperabilidad ya que el protocolo MQTT-SN es usado por los nodos sensores desplegados en la red y MQTT es utilizado por el servidor o bróker.

En el presente trabajo de titulación se desarrolla la implementación del diagrama de estados del convertidor de protocolos MQTT-SN a MQTT en una red de sensores mediante el lenguaje unificado de modelado (UML – Unified Modeling Language) [4], también se valida el diagrama de estados mediante un software de verificación con el objetivo de generar una referencia para futuros trabajos que fomenten el desarrollo de las redes de sensores en el ámbito de IoT.

Este documento tiene la siguiente estructura: en el capítulo 1 se mencionan los objetivos, el alcance y los fundamentos teóricos para diseñar el diagrama de estados del conversor de protocolos; en el capítulo 2 se definen los estados y procesos de los diagramas del convertidor de protocolos, se implementan los mismos utilizando UML y también en el software de validación UPPAAL; en el capítulo 3 se muestran los resultados de la validación del comportamiento del diagrama de estados y, por último, en el capítulo 4, se mencionan las conclusiones y recomendaciones obtenidas a lo largo del trabajo de integración curricular.

1.1 Objetivo general

El objetivo general de este trabajo de integración curricular es implementar el diagrama de estados del conversor de protocolos MQTT-SN a MQTT.

1.2 Objetivos específicos

Los objetivos específicos de este trabajo de integración curricular son:

1. Comprender la base teórica necesaria para el desarrollo del proyecto.

2. Describir el funcionamiento de los protocolos MQTT-SN y MQTT.

3. Elaborar el diagrama de estados del conversor de protocolos MQTT-SN a MQTT.

1.3 Alcance

En el presente trabajo, primero, se realiza una breve revisión teórica sobre las redes inalámbricas de sensores y conceptos de IoT. Después, se enfoca el estudio en los protocolos MQTT y MQTT-SN, se describe generalmente el principio de su funcionamiento, componentes y mensajes. Después de esta revisión de la teoría y operación de los protocolos, se desarrolla el diagrama de estados para la conversión del protocolo MQTT-SN a MQTT, proceso ejecutado en el nodo denominado Gateway.

El diseño del diagrama de estados para el Gateway tiene como requerimiento que el protocolo MQTT-SN opere sobre el nivel de enlace en una red de sensores o sea

encapsulado en el protocolo 802.15.4. Se considera que el Gateway que se implementará a futuro, usando este trabajo como referencia, es un nodo que permite la programación orientada a objetos y que maneje una interfaz 802.15.4. El diagrama de estados está desarrollado en UML para que se facilite su futura implementación en otros trabajos.

Para el diseño del diagrama de estados se considera un sistema conformado por un nodo cliente, un Gateway y un bróker, tal como se describe en la Figura 1, en donde el denominado Gateway realiza la conversión de protocolos. La consideración del cliente y del bróker en este trabajo se debe a que estos emiten los mensajes MQTT-SN y MQTT al Gateway. No forma parte de este trabajo de integración curricular la implementación del diagrama de estados del cliente ni del bróker.

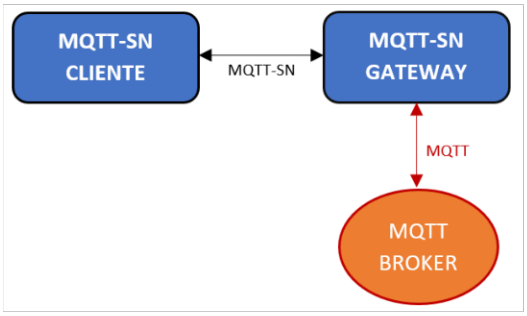


Figura 1: Esquema general de red considerado.

Para el desarrollo del diagrama de estados se utiliza UML, el cual provee de herramientas que permiten definir, visualizar y documentar los componentes de sistemas de telecomunicaciones, servicios, protocolos y sistemas de software [4], específicamente, permiten el diseño de diagramas de estado.

Como producto final del presente trabajo de integración curricular se presenta el diagrama de estados que representa el funcionamiento del conversor de protocolos MQTT-SN a MQTT. Se verifica el resultado del trabajo usando el software UPPAL Model Checker, el cual valida el comportamiento del diagrama de estados mediante la simulación de las transiciones entre estados basándose en el cumplimiento de condiciones definidas en el diseño del diagrama.

El resultado de este trabajo se podrá tomar como referencia para implementar el conversor de protocolos en cualquier lenguaje de programación orientado a objetos y ser compilado utilizando cualquier herramienta según la aplicación futura lo requiera. El alcance de este trabajo de integración curricular no incluye la implementación del código de los procesos que se realizan en cada estado para realizar la conversión de los protocolos.

1.4 Marco teórico

En esta sección se revisan brevemente los conceptos teóricos relacionados a las redes inalámbricas, estándar de comunicación 802.15.4, redes inalámbricas de sensores e internet de las cosas. Se hace énfasis en la revisión de los protocolos MQTT y MQTT-SN ya que estos son la base para el posterior desarrollo del diagrama de estados del conversor de protocolos.

1.4.1 Redes inalámbricas

Una red inalámbrica es un conjunto de dispositivos que se conectan mediante ondas de radio electromagnéticas con la finalidad de intercambiar información para un objetivo en específico, la interconexión de la red se consigue sin ningún tipo de medio físico guiado [5].

1.4.2 Estándar 802.15.4

Un estándar de comunicación define las normas para el intercambio de datos en una red. El estándar IEEE 802.15.4 define una comunicación de baja velocidad de datos entre dispositivos de mínimo consumo energético y en cortas distancias.
Este estándar define su operación en la banda de 2.4 GHz a nivel global, una velocidad de transmisión de 250 Kbps y un alcance de 100m sin repetidores [6]. También permite crear redes con topología de estrella y redes Ad-Hoc. Las redes Ad-Hoc tienen una comunicación descentralizada que no depende de una infraestructura previa con dispositivos de red, lo cual implica que no se requiere de un elemento de red que enrute todo el tráfico ya que cada dispositivo dirige la información que le llega hacia los demás [7].

1.4.3 Redes inalámbricas de sensores (Wireless sensor networks – WSN)

Las redes inalámbricas de sensores o WSN, se definen como una red de varios dispositivos electrónicos conectados inalámbricamente, estos se distribuyen a lo largo de un área física y están provistos de sensores con los que pueden monitorear un entorno y posteriormente transmitir la información recabada cumpliendo un objetivo específico para una aplicación [8]. Cada dispositivo es también conocido como nodo y es un elemento autónomo que recolecta información, la envía a un Gateway en donde se procesará y posteriormente se transmitirá a un dispositivo receptor o estación base donde se almacena o reproduce según la aplicación.

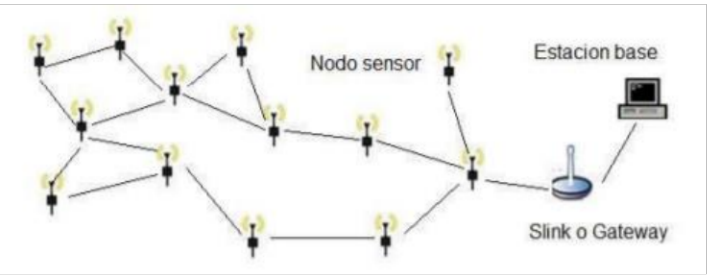


Figura 2: Red inalámbricas de sensores [8].

La información que los nodos recolectan puede variar dependiendo de la solución implementada con la WSN, pero principalmente se centran en medir variables físicas como temperatura, intensidad de iluminación, presión, vibración, humedad, etc.

Debido a sus limitados recursos tanto de procesamiento como energéticos, los nodos sensores de una WSN se encuentran una gran parte del tiempo en estado de reposo o también conocido como modo de bajo consumo de potencia. Es por esto que las redes inalámbricas de sensores necesitan protocolos que no sean muy exigentes en cuanto a los recursos de los nodos pero que les permitan transmitir información.

1.4.4 Internet de las cosas (Internet of things – IoT)

El internet de las cosas abre una etapa en la tecnología de comunicaciones debido a que introduce un nuevo contexto de conexión de elementos físicos a Internet, desde donde se posibilita el acceso remoto a la información y el control de sistemas a cualquier hora y desde cualquier lugar [9].

Para entender de mejor manera el concepto de IoT, se puede definirlo como una red conformada por dispositivos inteligentes o “cosas”, las cuales recogen y envían datos adquiridos mediante el monitoreo para cumplir una tarea o aplicación especifica.

Generalmente el sistema IoT está compuesto por:

- Cosas (Things): dispositivo u objeto que tiene alguna capacidad de procesamiento, puede recoger datos a través de sensores y además posee algún tipo de direccionamiento para ser identificado en la red IoT [9].
- Comunicaciones (Communication): referente a la infraestructura de red, sea esta cableada o inalámbrica, que transporta los datos obtenidos mediante sensores [9].
- Procesamiento y almacenamiento (Computing & Storage): Sistemas que analizan, procesan y almacenan datos para dar un servicio a las aplicaciones de IoT [9].
- Servicios y aplicaciones (Service & Applications): Soluciones que IoT ofrece para la interacción entre usuarios y aplicaciones específicas [9].

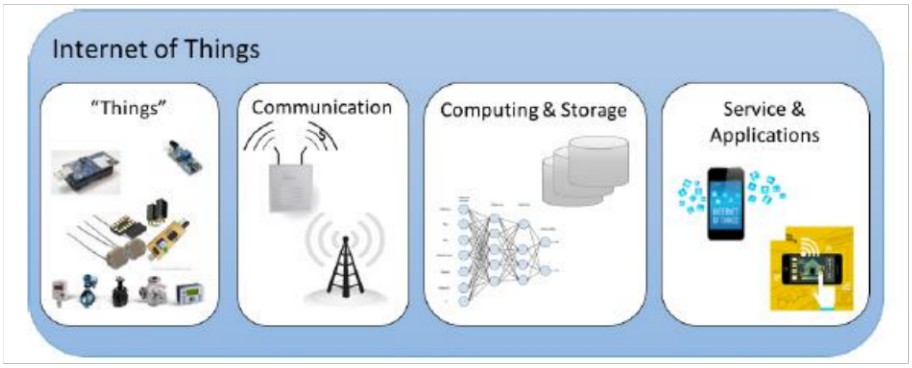


Figura 3: Diagrama de bloques de los componentes de un sistema IoT [9].

Los sistemas IoT integran una gran variedad de herramientas y tecnologías, por lo que se han desarrollado varias arquitecturas y protocolos que buscan garantizar una operación óptima teniendo en cuenta las restricciones de recursos de los dispositivos, su heterogeneidad y su compleja administración de red [10].

1.4.4.1 Protocolos para IoT

En la actualidad se pueden listar varios protocolos usados en aplicaciones de IoT, tales como CoAP (Constrained Application Protocol), MQTT (Message Queue Telemetry Transport), MQTT-SN (MQTT for Sensor Networks) y XMPP (Extensible Messaging and Presence Protocol), entre otros. Cada uno de ellos tiene características propias y una arquitectura de mensajería útil para diferentes tipos de aplicaciones de IoT, que tienen como premisa general utilizar efectivamente los recursos que usualmente son limitados en los nodos de red [11].

1.4.4.1.1 Protocolo MQTT (Message Queue Telemetry Transport)

MQTT (Message Queue Telemetry Transport) es el protocolo de transmision de mensajes que usa el modelo publicación/suscripción, está dirigido hacia dispositivos provistos de limitado ancho de banda. El protocolo trabaja sobre TCP/IP (Transmission Control Protocolo/Internet Protocol) u otros protocolos que permitan conexiones bidireccionales, ordenadas y sin pérdidas. El principal objetivo de MQTT es minimizar la necesidad de recursos energéticos y de ancho de banda para los dispositivos, manteniendo cierta fiabilidad en la red lo cual permite su uso en IoT o conexiones machine-to-machine (M2M) [12].

Su operación se basa en los siguientes componentes principales:

- Cliente: nodo que es desplegado en la red y puede cumplir como publicador o suscriptor [13].
    - Publicador: nodo cliente encargado de la transmisión de información hacia

el bróker sobre un “topic” determinado [13].

- Suscriptor: nodo cliente que recibe la información del bróker sobre un “topic” determinado [13].

- Bróker: es el componente central o servidor que actúa como intermediario encargándose de la recopilación de los datos que los nodos publicadores transmiten y después direcciona los mensajes a los nodos suscriptores que solicitan la información según el tipo de contenido o “topic” [13].
- Topic: es el tema al que los nodos clientes pueden suscribirse para recibir información [13].

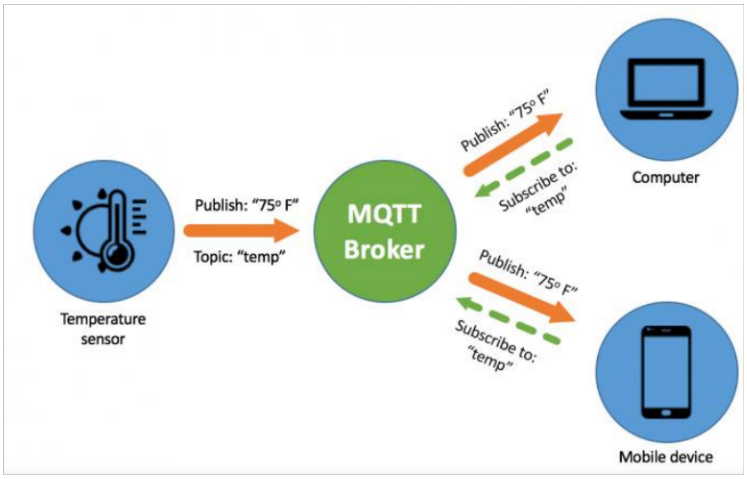


Figura 4: Modelo de comunicación y arquitectura MQTT [12].

En el protocolo MQTT se encuentra integrada la calidad de servicio o QoS (Quality of Service), este es el nivel de acuerdo entre transmisión y recepción para garantizar una entrega correcta de los mensajes. En el protocolo se definen tres niveles de QoS.

- QoS nivel 0: ofrece un servicio de “el mejor esfuerzo” para la entrega, lo que significa que la recepción del mensaje no está garantizada. El mensaje no se almacena después de su transmisión y no hay una confirmación de recepción por parte del bróker [13].
- QoS nivel 1: garantiza que el mensaje llegó al destino, al menos una vez, ya que éste se retransmite hasta que se reciba una confirmación o ACK por parte del bróker [13].
- QoS nivel 2: es el nivel más alto de calidad de servicio, garantiza que el mensaje llegó exactamente una sola vez mediante un intercambio de mensajes de control. Este método de control permite evitar la duplicación de mensajes por lo que en este nivel de QoS se aumenta el tiempo de procesamiento [13].

Para conseguir una comunicación entre los componentes de red, el protocolo MQTT posee una serie de mensajes de control por medio de los cuales se puede realizar el intercambio de información.

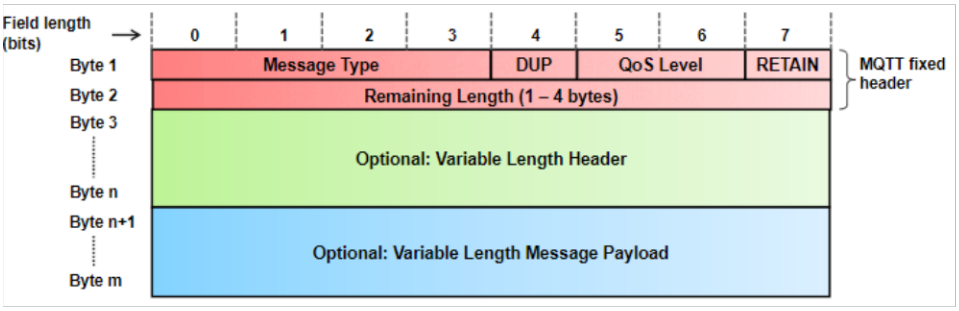


Figura 5: Formato de mensaje MQTT [14].

Como se puede observar en la Figura 5, un mensaje en el protocolo MQTT está compuesto por tres partes.

- Cabecera fija (Fixed header): es la parte inicial de un mensaje MQTT y es la única que está presente en todos los mensajes del protocolo [15]. Tiene entre sus campos:
    - "Message Type": indica el tipo de mensaje, como se detalla en la Tabla 1.1 [15].
    - "DUP": bit de duplicidad que ayuda al receptor a reconocer si ya recibió ese mensaje [15].
    - "QoS Level": indica el nivel de QoS [15].
    - "RETAIN": si está activo (1) indica al servidor que debe retener el mensaje [15].
    - "Remaining Length": indica los bytes restantes del mensaje incluyendo la cabecera variable y payload [15].
- Cabecera variable (Variable Header): no está incluida en todos los mensajes MQTT, en el mayor de los casos contiene los bytes correspondientes a la identificación del mensaje (Packet Identifier) o del tema (Topic) [15].
- Carga útil (Payload): contiene la información que se va a transmitir, no está presente en todos los mensajes como se muestra en la Tabla 1.1 [15].

Tabla 1.1 Descripción mensajes MQTT [15].

Valores de Message Type (Dec)	Nombre	Dirección de transmisión	Descripción	Payload
1	CONNECT	Cliente→Servidor	Petición de conexión	Requerido
2	CONNACK	Cliente←Servidor	Confirmación de conexión	Ninguno
3	PUBLISH	Cliente←→Servidor	Mensaje de publicación	Opcional
4	PUBACK	Cliente←→Servidor	Confirmación publicación	Ninguno
5	PUBREC	Cliente←→Servidor	Recepción de publicación (entrega asegurada I)	Ninguno
6	PUBREL	Cliente←→Servidor	Lanzamiento de publicación (entrega asegurada II)	Ninguno
7	PUBCOMP	Cliente←→Servidor	Publicación completada (entrega asegurada III)	Ninguno
8	SUBSCRIBE	Cliente→Servidor	Petición de suscripción	Requerido
9	SUBACK	Cliente←Servidor	Confirmación suscripción	Requerido
10	UNSUBSCRIBE	Cliente→Servidor	Petición de cancelación de suscripción	Requerido
11	UNSUBACK	Cliente←Servidor	Confirmación cancelación de suscripción	Ninguno
12	PINGREQ	Cliente→Servidor	Solicitud de PING	Ninguno
13	PINGRESP	Cliente←Servidor	Respuesta de PING	Ninguno
14	DISCONNECT	Cliente→Servidor	Cliente desconectado	Ninguno

Los tres niveles de QoS definidos en el protocolo MQTT utilizan los mensajes de la Tabla 1.1 para realizar el siguiente intercambio de mensajes:

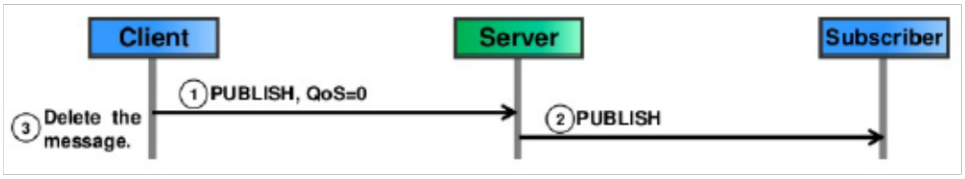


Figura 6: Flujo de mensajes con QoS nivel 0 [14].

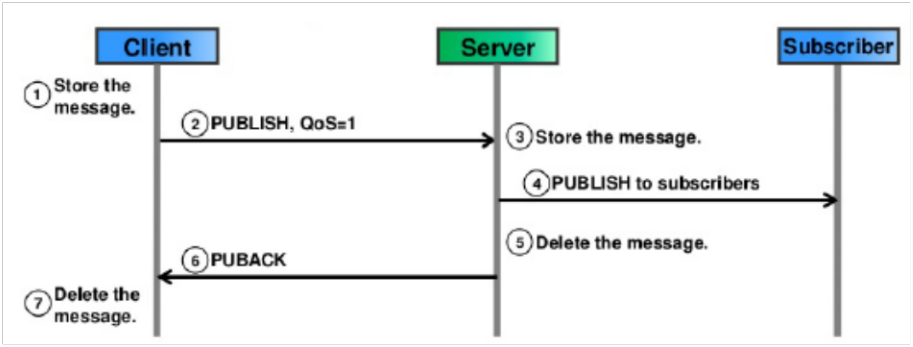


Figura 7: Flujo de mensajes con QoS nivel 1 [14].

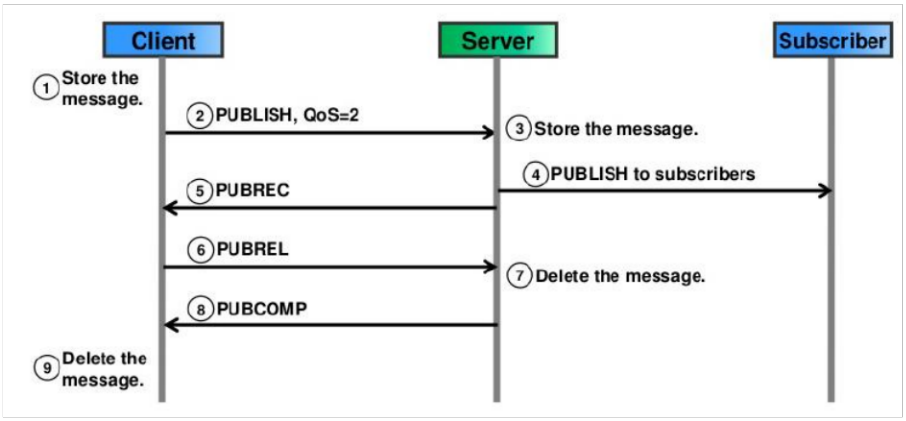


Figura 8: Flujo de mensajes con QoS nivel 2 [14].

1.4.4.1.2 Protocolo MQTT-SN (MQTT – Sensor Networks)

El protocolo MQTT-SN es una extensión de MQTT, también es un protocolo de publicación/suscripción, pero su uso está orientado a redes inalámbricas de sensores (WSN). El objetivo de MQTT-SN es extender el alcance de MQTT más allá de la infraestructura TCP/IP ya que en este protocolo se utiliza UDP (User Datagram Protocol) para que los dispositivos de red no necesiten una conexión permanente [16].

MQTT-SN es considerado como un protocolo ligero ya que está adaptado a las limitaciones de los nodos sensores, por lo que se trabaja con un bajo ancho de banda, longitud de mensajes cortos, bajo consumo de batería y con limitada capacidad de procesamiento y almacenamiento [16].

En este protocolo tenemos los siguientes elementos principales para su funcionamiento:

- Cliente: son los nodos que se encuentran en la WSN y al igual que en MQTT pueden operar como publicador o suscriptor. Para poder comunicarse con un bróker MQTT requieren un Gateway [2].
- Gateway: Sirve de enlace entre los nodos cliente que utilizan el protocolo MQTT-SN y requieren comunicarse con el servidor o bróker que usa el protocolo MQTT por lo que este nodo es responsable de la conversión de protocolos [2].

- Forwarder: es un nodo encargado de reencapsular las tramas MQTT-SN en otras tramas iguales, no realizan ningún cambio y las envían directamente al Gateway MQTT-SN para que se realice la conversión de protocolos y se envíe al servidor [2].

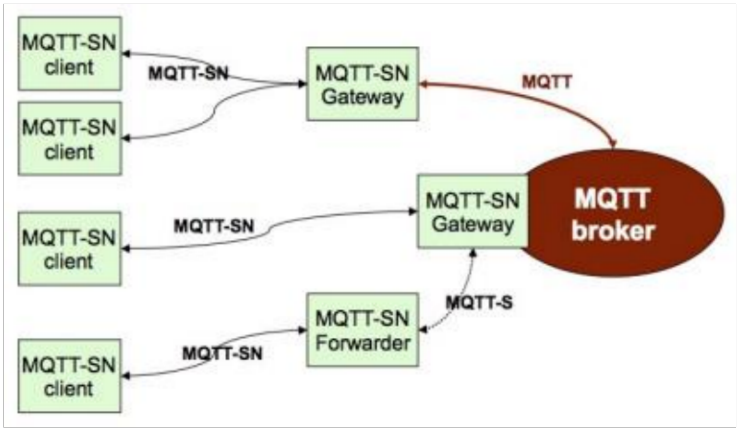


Figura 9: Arquitectura MQTT-SN [2].

El nodo Gateway MQTT-SN puede tener dos roles en una arquitectura:

- Gateway transparente: el nodo administra una puerta de enlace para cada cliente por lo que la conversión de protocolos se realiza individualmente para cada flujo MQTT-SN generado en la WSN [2].
- Gateway agregado: el nodo posee solo una conexión MQTT con el bróker y administra la misma para un grupo de nodos cliente. Manejar una arquitectura con Gateway agregado es complejo debido a que se debe mantener conexiones simultaneas con el cliente [2].

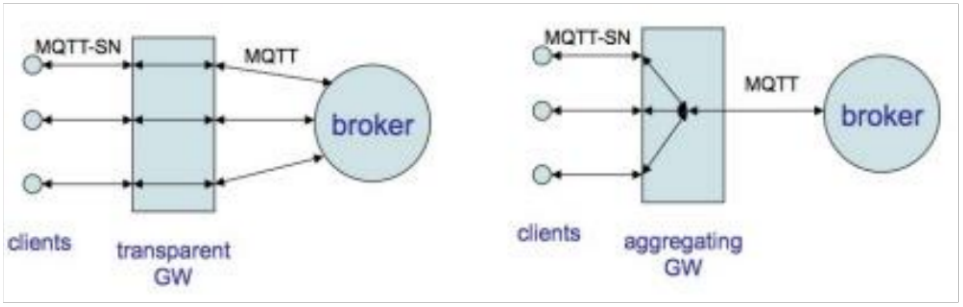


Figura 10: Gateway MQTT-SN transparente y agregado [2].

Al igual que en el anterior protocolo, MQTT-SN tiene mensajes de control que son utilizados por los nodos clientes para establecer una comunicación con el Gateway. El formato del mensaje varía con respecto al protocolo MQTT, tal como se muestra a continuación.

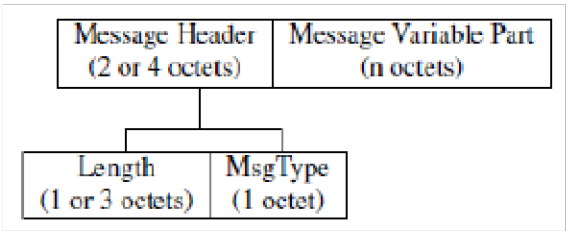


Figura 11: Formato de mensaje MQTT-SN [2].

En la Figura 11 se presentan los componentes de un mensaje MQTT-SN los cuales son:

- Cabecera de mensaje (Message Header): es la parte inicial del mensaje, está compuesta de dos elementos.
    - “Length”: especifica el número total de bytes contenidos en el mensaje [2].
    - “MsgType”: indica el tipo de mensaje, como se detalla en la Tabla 1.2 [2].

Tabla 1.2 Descripción mensajes MQTT-SN [2].

Valores de MsgType (Hex)	Nombre	Dirección de transmisión	Descripción
0x00	ADVERTISE	Cliente←Gateway	Búsqueda de clientes en broadcast
0x01	SEARCHGW	Cliente→Gateway	Búsqueda de GW en broadcast
0x02	GWINFO	Cliente←Gateway	Respuesta a búsqueda de GW
0x04	CONNECT	Cliente→ Gateway	Petición de conexión
0x05	CONNACK	Cliente← Gateway	Confirmación de conexión
0x06	WILLTOPICREQ	Cliente← Gateway	Petición de Will topic
0x07	WILLTOPIC	Cliente→ Gateway	Respuesta a petición de Will topic
0x08	WILLMSGREQ	Cliente← Gateway	Petición de Will message
0x09	WILLMSG	Cliente→ Gateway	Respuesta a petición de Will message
0x0A	REGISTER	Cliente←→Gateway	Información de topic id
0x0B	REGACK	Cliente←→Gateway	Confirmación de información de topic id
0x0C	PUBLISH	Cliente←→Gateway	Mensaje de publicación
0x0D	PUBACK	Cliente←→ Gateway	Confirmación publicación
0x0E	PUBCOMP	Gateway←→Servidor MQTT	Publicación completada (QoS-2)
0x0F	PUBREC	Gateway ←→Servidor MQTT	Recepción de publicación (QoS-2)
0x10	PUBREL	Gateway ←→Servidor MQTT	Lanzamiento de publicación (QoS-2)
0x12	SUBSCRIBE	Cliente→Servidor MQTT	Petición de suscripción
0x13	SUBACK	Cliente←Servidor MQTT	Confirmación suscripción
0x14	UNSUBSCRIBE	Cliente→Servidor MQTT	Petición de cancelación de suscripción
0x15	UNSUBACK	Cliente←Servidor MQTT	Confirmación cancelación de suscripción
0x16	PINGREQ	Cliente←→Gateway	Solicitud de PING

0x017	PINGRESP	Cliente←→Gateway	Respuesta de PING
0x18	DISCONNECT	Cliente→Gateway	Cliente desconectado
0x1A	WILLTOPICUPD	Cliente→Gateway	Actualización Will topic
0x1B	WILLTOPICRESP	Cliente←Gateway	Confirmación actualización Will topic
0x1C	WILLMSGUPD	Cliente→Gateway	Actualización Will message
0x1D	WILLMSGRESP	Cliente←Gateway	Confirmación actualización Will message

- Parte variable de mensaje (Message Variable Part): el contenido de este componente depende del tipo de mensaje MQTT-SN [2]. Los campos definidos para la parte variable de mensaje son los siguientes.
    - ClientId: identificación de cada cliente para con el bróker [2].
    - Data: es el payload del mensaje a publicar [2].
    - Duration: especifica la duración del mensaje en segundos [2].
    - Flags: contiene una serie de banderas útiles para el protocolo como se muestra en la Figura 12 [2].
        - DUP: bit de duplicidad que ayuda al receptor a reconocer si ya recibió ese mensaje [2].
        - QoS: al igual que en MQTT, indica el nivel de QoS [2].
        - Retain: si está activo (1) indica al servidor que debe retener el mensaje [2].
        - Will: indica si el cliente solicita un tema (Will topic) o un mensaje (Will message) [2].
        - CleanSession: indica si se debe mantener la sesión de comunicación entre cliente y servidor [2].
        - TopicIdType: indica si el campo TopicId o TopicName incluido en este mensaje contiene un “topic id” normal, predefinido o “topic name” corto [2].

DUP (bit 7)	QoS (6,5)	Retain (4)	Will (3)	CleanSession (2)	TopicIdType (1,0)

Figura 12: Campo Flags [2].

    - GwAdd: contiene la dirección de un Gateway en la red [2].
    - GwId: contiene el identificador del Gateway en la red [2].
    - MsgId: es el identificador de mensaje que se va a transmitir [2].
    - ProtocolId: únicamente incluido en el mensaje CONNECT, indica la versión de protocolo [2].
    - Radius: indica el radio del broadcast a los nodos en la red [2].

- ReturnCode: indica si un mensaje fue aceptado, rechazado por congestión, por Topic Id invalido o por no ser soportado [2].
- TopicId: contiene el identificador del topic [2].
- TopicName: contiene el nombre del topic [2].
- WillMsg: contiene el Will message [2].
- WillTopic: contiene el Will topic [2].

1.4.5 Lenguaje unificado de modelado (UML – Unified Modeling Language)

Es un lenguaje de modelado que permite definir, visualizar y documentar los componentes de sistemas de telecomunicaciones, servicios, protocolos y sistemas de software [4]. UML permite representar el comportamiento de un sistema a través del modelado de objetos discretos concatenados que interactúan entre sí por medio de acciones y eventos.

UML es reconocido como un lenguaje de propósito general ya que no está ligado a ningún método de implementación específico, por lo que puede ser usado para crear una referencia de futuros desarrollos en cualquier método orientado a objetos.

Uno de los objetivos del UML es mantener la simplicidad en su desarrollo, abarcando todos los aspectos fundamentales del comportamiento de un sistema, pero sin profundizar en detalles que compliquen el modelamiento base que se pretende generar.

En el lenguaje unificado de modelado las bases conceptuales se engloban en “vistas”, una vista es un subconjunto de UML que representa un aspecto de un sistema [17]. Las vistas se dividen en las siguientes áreas:

- Clasificación estructural: describe los elementos del sistema y sus relaciones con otros elementos [17].
- Comportamiento dinámico: describe el comportamiento de un sistema en el tiempo [17].
- Gestión del modelo: describe la organización de los propios modelos en unidades jerárquicas [17].

1.4.5.1 Diagramas de estado

El diagrama de estado pertenece al área de comportamiento dinámica definida en UML, muestra un conjunto de estados por los cuales pasa un objeto en un momento dado. Cada estado está conectado por transiciones, las cuales ocurren cuando se satisfacen ciertas condiciones o en respuesta a eventos (mensajes recibidos, tiempo rebasado, errores, etc.) [18].

Un diagrama de estado generalmente está compuesto por:

- Estado: identifica la naturaleza en el periodo de tiempo actual de un objeto donde

espera alguna operación. Se representa por un rectángulo con bordes redondos, puede contener el nombre de estado y acciones que se realizan a la entrada, salida o durante el estado (entry, exit o do) [19].

- Transición: se simboliza con una flecha entre estados [19].
- Eventos: instancia que ocasiona una transición de estado. Su aparición puede estar ligada al cumplimiento de una condición, recepción de un mensaje o expiración de un periodo de tiempo. El evento puede ir etiquetado encima de la flecha de transición [19].
- Acción: es una operación automática que no se puede interrumpir por un evento y se ejecuta hasta su finalización [18]. Se simboliza con un / previo a la descripción de la acción.
- Subestado: estado contenido dentro de otro, con transiciones entre ellos y conexiones a nivel superior [19].
- Estado compuesto: estado que contiene subestados anidados.
- Primer estado: indicador para el primer estado en el proceso, se representa por un círculo negro con una flecha de transición.
- Terminador: indicador de proceso terminado, se representa con un punto en el interior de un círculo.
- Pseudoestado de opción: indicador que implica resultados con ramificaciones de decisión, se representa con un diamante.

1.4.6 UPPAL Model Checker

UPPAL es un software que integra un conjunto de herramientas para modelar y validar sistemas en tiempo real mediante simulación gráfica. Fue desarrollado en conjunto por el departamento de “Tecnologías de la Información” de la Universidad Uppsala en Suecia y el departamento de “Ciencias de la Computación” de la Universidad Aalborg en Dinamarca [20].

UPPAL permite verificar el comportamiento de un sistema ya que está provisto de una interfaz gráfica que permite implementar el diagrama de estados y sus transiciones. Además cada transición es considerada como un canal al cual se le puede asignar un nombre y una condición de ocurrencia, esta puede ser definida por el valor de una variable de tipo entera o booleana.

A parte de la interfaz gráfica, este software tiene un motor de simulación desde el cual se pueden evaluar las posibles transiciones entre estados según el cumplimiento de condiciones definidas previamente y la lógica del flujo de transiciones entre estados del diagrama creado. En esta interfaz se evalúan las transiciones entre estados para determinar la validez del diagrama o caso contrario encontrar inconsistencias en el comportamiento del mismo que puedan perjudicar una futura implementación.

2 METODOLOGÍA

En este capítulo se presentan los diseños de diagrama de estados del convertidor de protocolo MQTT-SN a MQTT y viceversa utilizando UML, se definen los estados creados y se analizan los procesos definidos en cada uno. Además, se presenta el diseño de los diagramas en el software UPPAAL para verificar su comportamiento.

2.1 Conversión de protocolo MQTT-SN a MQTT

Como se explicó anteriormente, en una WSN los nodos sensores tienen capacidades limitadas y por esta razón utilizan el protocolo MQTT-SN para transmitir datos. Los nodos de red requieren establecer una comunicación con el servidor/bróker ya que este es el encargado de recibir, guardar y reproducir la información adquirida. Debido a que el bróker utiliza el protocolo MQTT, el nodo Gateway es el componente que funciona como interfaz entre ambos protocolos y realiza la conversión de MQTT-SN a MQTT.

Dado que MQTT-SN es una extensión de MQTT para redes inalámbricas de sensores, ambos protocolos guardan una similitud en sus mensajes de control, pero al no poseer las mismas capacidades de transmisión tienen estructuras de mensaje diferentes tal como se evidencia en la Figura 5 y Figura 11.

Debido a las diferentes estructuras de mensaje en cada protocolo, el nodo sensor debe establecer en primer lugar una comunicación con el Gateway para que este adapte los mensajes de control y puedan ser comprendidos por el bróker, siendo este el proceso de conversión de protocolos.

En la Figura 13 se observa el diagrama de estados diseñado en este trabajo, el cual representa el funcionamiento del convertidor de protocolos.

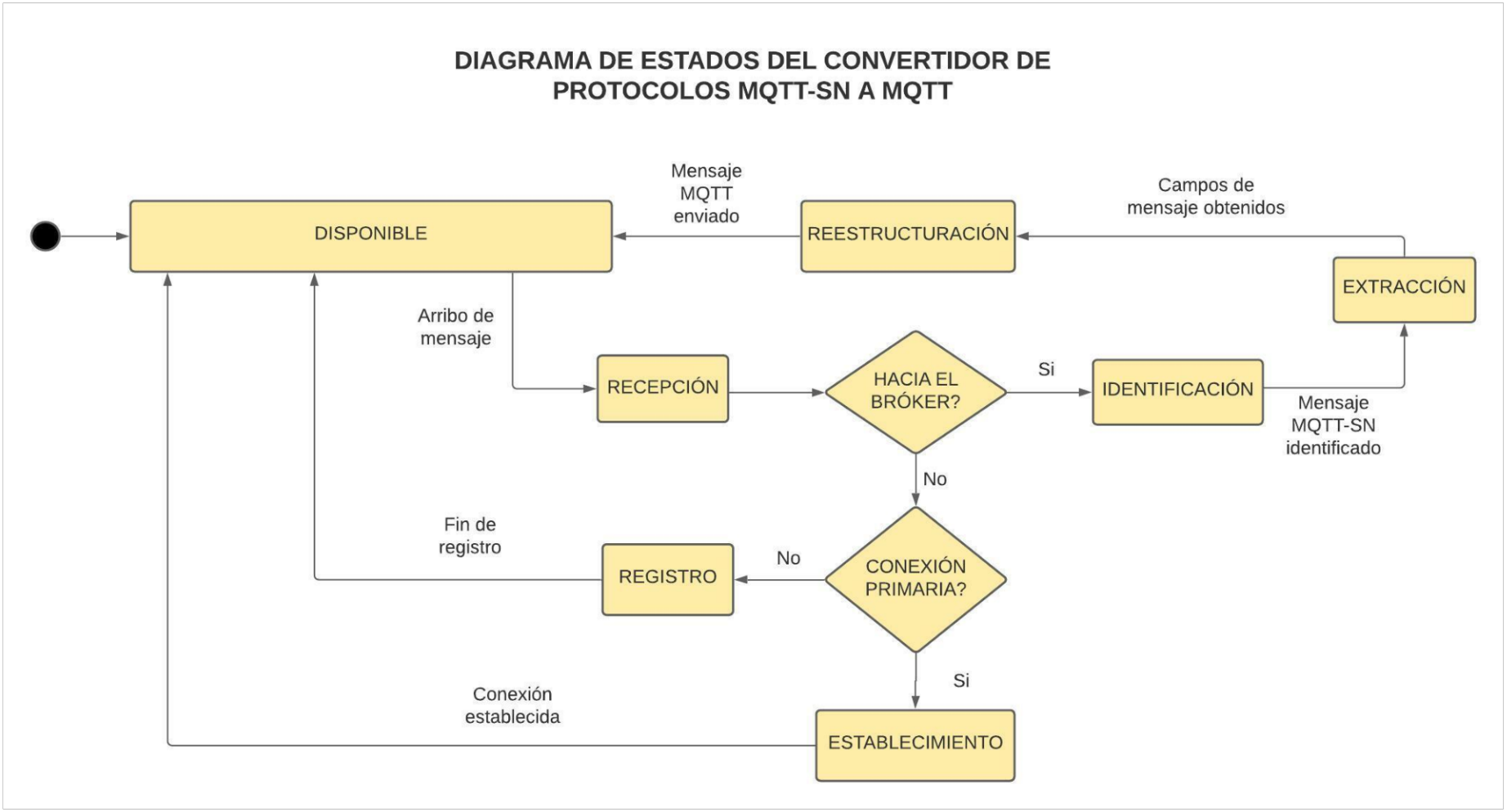


Figura 13: Diagrama de estados del convertidor de protocolos MQTT-SN a MQTT.

Para la elaboración del diagrama de la Figura 13 se crearon diferentes estados, cada cual implica uno o más procesos que contribuyen a la funcionalidad del convertidor de protocolos por lo que se procede a explicar cada uno.

DISPONIBLE

Es el estado principal del diagrama, en el cual el nodo Gateway acepta mensajes provenientes de la red y está apto para establecer una comunicación. Si se presenta un evento de arribo de mensaje se procede a realizar una transición al estado de RECEPCIÓN.

Se considera a este estado como base del desarrollo del diagrama del convertidor de protocolos ya que al finalizar los procesos en los estados de REGISTRO, ESTABLECIMIENTO y REESTRUCTURACIÓN se realiza una transición de vuelta hacia el estado DISPONIBLE.

RECEPCIÓN

Cuando el Gateway se encuentra en estado DISPONIBLE y se presenta el evento de arribo de mensaje, se realiza la transición al estado RECEPCIÓN. En el presente estado el Gateway recibe el mensaje y decide si es un mensaje dirigido al bróker MQTT o si es dirigido a sí mismo para establecer una conexión con un nodo sensor.

Si el mensaje está dirigido hacia el bróker se realiza una transición hacia el estado IDENTIFICACIÓN, caso contrario se cambia al siguiente pseudoestado de opción que divide las conexiones nodo sensor – Gateway en los estados de ESTABLECIMIENTO y REGISTRO. Ambos estados implican un intercambio de mensajes que será profundizado en el análisis individual de cada uno.

ESTABLECIMIENTO

En este estado se realiza la conexión primaria con el nodo sensor, ya que previo a cualquier transmisión hacia el bróker se debe establecer una conexión con el Gateway tal como se muestra en la Figura 14.

Figura 14: Proceso de conexión [2].

El proceso inicia con un mensaje CONNECT enviado por el cliente o nodo sensor hacia el

Gateway, en la Figura 15 se muestra la estructura del mensaje.

Length (octet 0)	MsgType (1)	Flags (2)	ProtocolId (3)	Duration (4,5)	ClientId (6:n)

Figura 15: Mensaje CONNECT [2].

En este caso, el mensaje tiene los siguientes campos:

- Length: indica la cantidad total de bytes contenidos en el mensaje.
- MsgType: identificador de mensaje CONNECT (0x04).
- Flags: la bandera Will está activa (1), indica que el cliente desea enviar su indicador de topic (Will topic) y su indicador de mensaje (Will Message).
- ProtocolId: usado únicamente en el mensaje CONNECT y fijado en el valor 0x01.
- Duration: contiene el valor de Keep Alive Timer.
- ClientId: contiene el identificador del cliente.

El Gateway recibe el mensaje, detecta que se trata de una nueva conexión y analiza las banderas del campo Flags por lo que procede a enviar el mensaje WILLTOPICREQ para indicar al cliente que envíe su Will topic, la estructura del mensaje se observa en la Figura 16.

Length (octet 0)	MsgType (1)

Figura 16: Mensaje WILLTOPICREQ [2].

- Length: indica la cantidad total de bytes contenidos en el mensaje.
- MsgType: identificador de mensaje WILLTOPICREQ (0x06).

Posteriormente del lado del cliente se envía el mensaje WILLTOPIC, el cual tiene la siguiente estructura.

Length (octet 0)	MsgType (1)	Flags (2)	WillTopic (3:n)

Figura 17: Mensaje WILLTOPIC [2].

- Length: indica la cantidad total de bytes contenidos en el mensaje.
- MsgType: identificador de mensaje WILLTOPIC (0x07).
- Flags: las banderas QoS y Retain están activas indicando el Will QoS y el Will Retain.
- WillTopic: contiene el nombre del Will topic definido por el nodo sensor.

Una vez recibido el mensaje WILLTOPIC, el Gateway envía el mensaje WILLMSGREQ, con la misma estructura de la Figura 16 pero con el campo MsgType en 0x08. Con este mensaje se solicita al nodo cliente que envíe su Will message con el mensaje WILLMSG, el cual tiene la siguiente estructura.

Length (octet 0)	MsgType (1)	WillMsg (2:n)

Figura 18: Mensaje WILLMSG [2].

- Length: indica la cantidad total de bytes contenidos en el mensaje.
- MsgType: identificador de mensaje WILLMSG (0x09).
- WillMsg: contiene el nombre del Will message definido por el nodo sensor.

Como respuesta a este último mensaje, el Gateway envía un mensaje CONNACK para indicar que el proceso de establecimiento de conexión está completo y el nodo cliente ya puede comunicarse con el bróker MQTT.

La estructura y campos del mensaje CONNACK se muestra en la Figura 19.

Length (octet 0)	MsgType (1)	ReturnCode (2)

Figura 19: Mensaje CONNACK MQTT-SN [2].

- Length: indica la cantidad total de bytes contenidos en el mensaje.
- MsgType: identificador de mensaje CONNACK (0x05).
- ReturnCode: contiene el valor 0x00 para indicar que la conexión fue aceptada.

En el diagrama de estados diseñado, al culminar el proceso del estado ESTABLECIMIENTO se realiza una transición hacia el estado DISPONIBLE para que el convertidor de protocolos pueda continuar con su operación.

REGISTRO

Después de que el nodo cliente haya establecido una conexión primaria con el Gateway, este ya puede comunicarse con el bróker MQTT, pero antes debe realizar un proceso de registro en el presente estado.

Debido al limitado ancho de banda característico de las WSN, no se puede enviar la información junto con el nombre de topic tal cual se hace en MQTT. Es por esto que en el protocolo MQTT-SN se desarrolló el registro de “topic name”, en este proceso se reemplaza el nombre de topic por un identificador más corto “topicId”.

Para iniciar el registro, el cliente envía un mensaje REGISTER al Gateway, el cual tiene la siguiente estructura.

Length (octet 0)	MsgType (1)	TopicId (2,3)	MsgId (4:5)	TopicName (6:n)

Figura 20: Mensaje REGISTER [2].

- Length: indica la cantidad total de bytes contenidos en el mensaje.
- MsgType: identificador de mensaje REGISTER (0x0A).
- TopicId: contiene el valor 0x0000 debido a que el identificador de topic es asignado por el Gateway en la respuesta a este mensaje.
- MsgId: creado por el nodo sensor para identificar el correspondiente ACK que contiene el TopicId.
- TopicName: contiene el nombre de topic al cual se le asignara un identificador.

Como respuesta a REGISTER, el Gateway genera el mensaje REGACK el cual tiene la

siguiente estructura y campos.

Length (octet 0)	MsgType (1)	TopicId (2,3)	MsgId (4,5)	ReturnCode (6)

Figura 21: Mensaje REGACK [2].

- Length: indica la cantidad total de bytes contenidos en el mensaje.
- MsgType: identificador de mensaje REGACK (0x0B).
- TopicId: contiene el valor que deberá ser usado como topic id en una futura transmisión de información mediante el mensaje PUBLISH.
- MsgId: es el valor recibido en el mensaje REGISTER.
- ReturnCode: contiene el valor 0x00 para indicar que el registro fue exitoso.

Una vez culminado este proceso, el nodo sensor ya es apto para publicar información hacia el bróker MQTT y que este a su vez la almacene o reproduzca según la aplicación desarrollada.

Después de haber analizado los procesos realizados en los estados ESTABLECIMIENTO y REGISTRO, se puede entender su importancia en el proceso de conversión de protocolos.

En el diagrama de estados, tras haber finalizado el registro del nodo cliente, se realiza una transición al estado DISPONIBLE y se espera el arribo de mensajes dirigidos hacia el bróker.

IDENTIFICACIÓN

Si el nodo cliente desplegado en la WSN ya realizó su correspondiente conexión primaria con el Gateway y también registró su topic name, ya tiene disponible la comunicación con el bróker usando el nodo Gateway como intermediario, el cual está encargado de la adaptación de mensajes entre los protocolos MQTT-SN y MQTT.

Cuando en el estado RECEPCIÓN se establece que el mensaje recibido en el Gateway es dirigido al bróker se realiza la transición al estado de IDENTIFICACIÓN. En el presente estado se analiza la cabecera del mensaje recibido, específicamente el campo MsgType, el cual contiene un código de identificación que corresponde a alguno de los mensajes definidos en el protocolo MQTT-SN, tal como se puede evidenciar en la Tabla 1.2.

Después de analizar el campo MsgType se define el mensaje recibido y se procede a realizar una transición al estado EXTRACCIÓN.

EXTRACCIÓN

Después del proceso de identificación del mensaje MQTT-SN, realizado en el estado anterior, se procede al estado denominado EXTRACCIÓN. En este se analiza la salida del estado IDENTIFICACIÓN para conocer el mensaje MQTT-SN transferido al Gateway. Se utiliza la estructura del mensaje para tomar campos específicos que serán usados para la creación del mensaje MQTT en el estado siguiente. Una vez terminado el proceso de toma

de datos del mensaje se realiza una transición al estado REESTRUCTURACIÓN.

REESTRUCTURACIÓN

En este estado se reciben los campos del mensaje MQTT-SN que se va a enviar al bróker para hacer la adaptación hacia el mensaje MQTT y así pueda ser procesado en el destino. En REESTRUCTURACIÓN el procedimiento que se realiza es una asignación de un campo MQTT-SN en el correspondiente campo MQTT, para esto en algunos casos se requiere información previa del nodo sensor almacenada en el Gateway, gracias a la comunicación previa que se debe realizar con el bróker.

Después del proceso de adaptación y generación del mensaje MQTT realizado en el presente estado, se realiza el envío del mensaje hacia el bróker y con este evento se produce una transición hacia el estado base DISPONIBLE.

Una vez definidos los estados de IDENTIFICACIÓN, EXTRACCIÓN y REESTRUCTURACIÓN, es importante aclarar que la conversión realizada en el Gateway depende del mensaje de control enviado por el nodo sensor.

A continuación se presentan las estructuras del mensaje MQTT-SN recibido y del mensaje MQTT transferido, un esquema de la correspondencia de campos en la conversión de cada mensaje y los procesos realizados en los últimos 3 estados del diagrama desarrollado en este trabajo.

- CONNECT

Length (octet 0)	MsgType (1)	Flags (2)	ProtocolId (3)	Duration (4,5)	ClientId (6:n)

Figura 22: Mensaje CONNECT MQTT-SN [2].

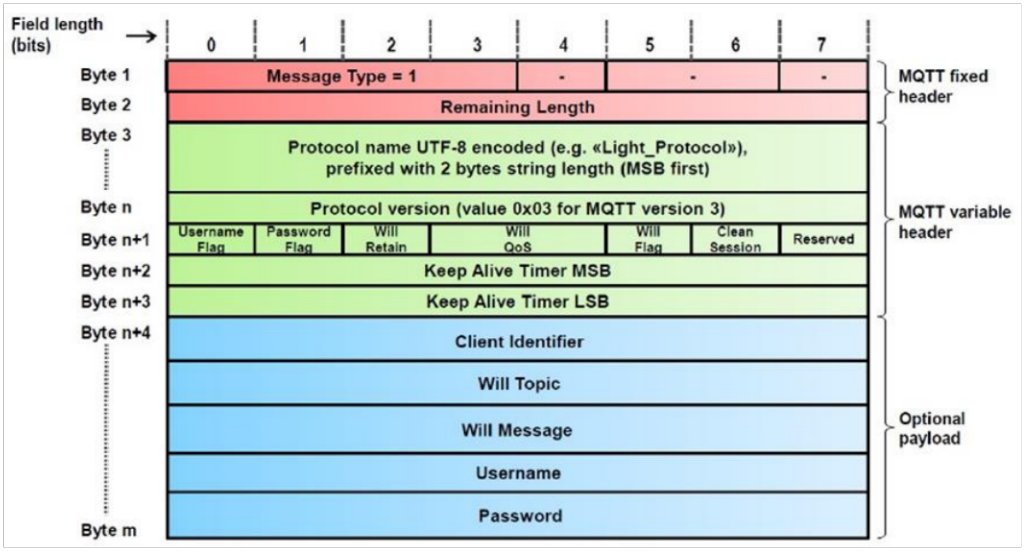


Figura 23: Mensaje CONNECT MQTT [14].

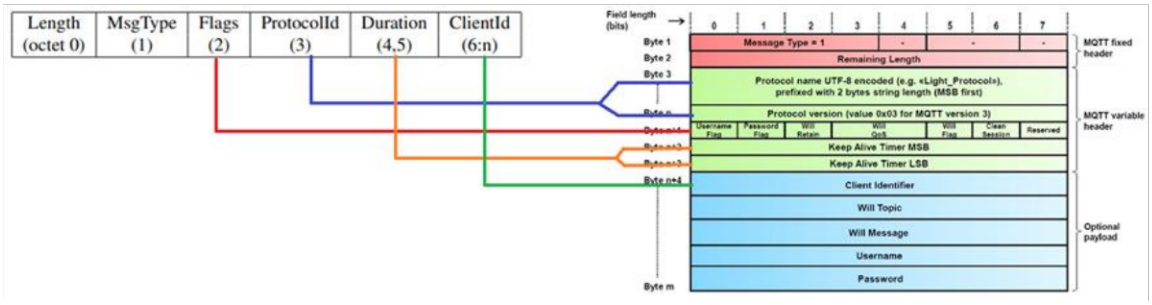


Figura 24: Esquema de correspondencia de campos.

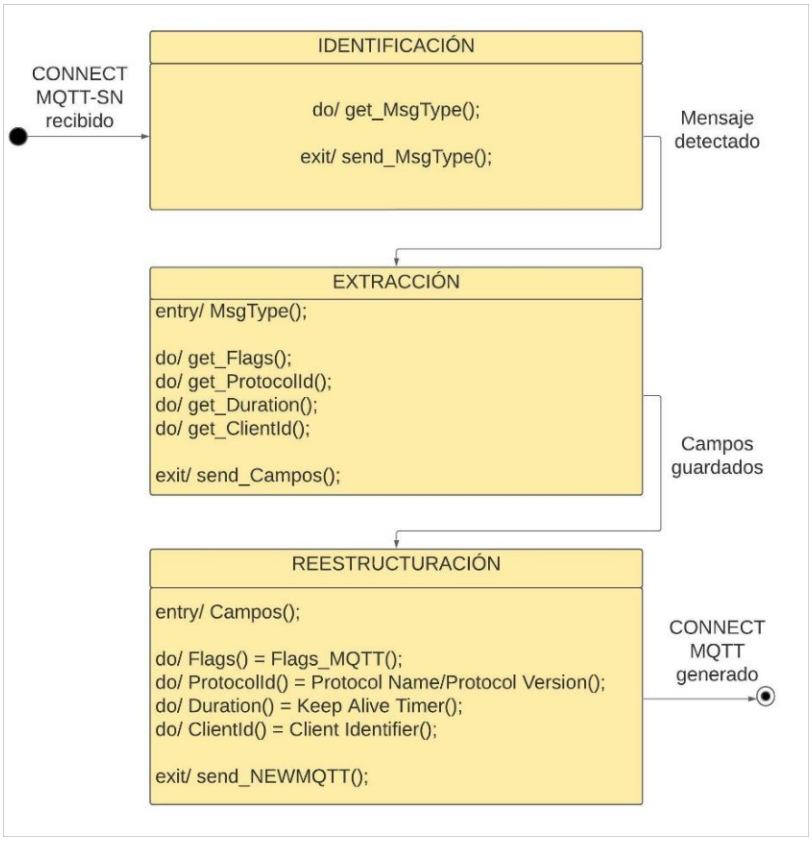


Figura 25: Proceso de conversión MQTT-SN a MQTT.

En la Figura 24 se puede observar la correspondencia de campos en la conversión de protocolos del mensaje CONNECT, usando este esquema como referencia se puede analizar los procesos realizados en el diagrama de estados, como se muestra en la Figura 25.

En el estado IDENTIFICACIÓN se analiza el mensaje recibido en el Gateway, en la cabecera de este se ubica el campo MsgType que contiene el identificador del mensaje CONNECT (0X04) y se envía en una acción de salida hacia el siguiente estado.

En el estado EXTRACCIÓN se recibe el valor de MsgType en una acción de entrada, conociendo el mensaje MQTT-SN se procede a ejecutar el proceso de adquisición de los campos Flags, ProtocolId, Duration y ClientId. Estos son almacenados en la variable "Campos()", creada para agrupar los valores objetivo del proceso de EXTRACCIÓN y enviar el conjunto en una acción de salida al siguiente estado.

En el estado REESTRUCTURACIÓN se ingresa la variable con la información de los campos necesarios para crear el mensaje CONNECT MQTT que posteriormente será enviado al bróker. En este estado se realizan las siguientes asignaciones:

Los valores de Length y MsgType son generados en el proceso de conversión debido a que no se mantienen iguales entre los protocolos. El primero será el total de bytes del mensaje y el segundo se generará según los valores de la Tabla 1.1, siendo un número binario de 4 bits el que será enviado en el nuevo mensaje.

- Flags:

Las banderas DUP, QoS, Retain y TopicIdType no se usan en este caso por lo que no son tomadas en cuenta para la asignación.

Bandera Will, si está activa (1) se realiza la misma asignación en el campo Will Flag del mensaje MQTT, pero los valores de los campos Will topic y Will Message son ingresados por el Gateway, que previamente en el estado ESTABLECIMIENTO recibió esos datos por parte del cliente mediante los mensajes WILLTOPIC y WILLMSG.

Clean Session, se realiza la misma asignación al campo Clean Session del mensaje MQTT, si está activo se borra la sesión.

- ProtocolId:

En el protocolo MQTT-SN, solo el mensaje CONNECT tiene este campo y por defecto tiene asignado el valor 0x01. En el caso del protocolo MQTT, de igual manera el mensaje CONNECT es el único que tiene los campos “Protocolo Name” y “Protocol Version” los cuales también tienen valores ya asignados, el primero compuesto por 6 bytes que codifican “MQTT” y el segundo tiene asignado el valor 0x03 para indicar la versión del protocolo. Dado que sólo en este caso los valores ya están definidos para un mensaje específico, en el diagrama de estados se muestra como si fuera un proceso de asignación de campos correspondientes.

- Duration:

Este campo contiene el valor de Keep Alive Timer del mensaje MQTT por lo que se realiza la asignación del mismo valor.

- ClientId:

En ambos protocolos, este campo tiene el mismo valor de 1 a 23 caracteres que identifica al cliente. Por lo que se realiza la asignación del mismo valor en el campo “Client Identifier”.

Después de realizar todas las asignaciones correspondientes se envía el mensaje MQTT, que dentro del estado se denomina NEWMQTT, se da por finalizado el proceso y se realiza la transición al estado DISPONIBLE como se observa en el diagrama de estados desarrollado.

- PUBLISH

Length (octet 0)	MsgType (1)	Flags (2)	TopicId (3-4)	MsgId (5-6)	Data (7:n)

Figura 26: Mensaje PUBLISH MQTT-SN [2].

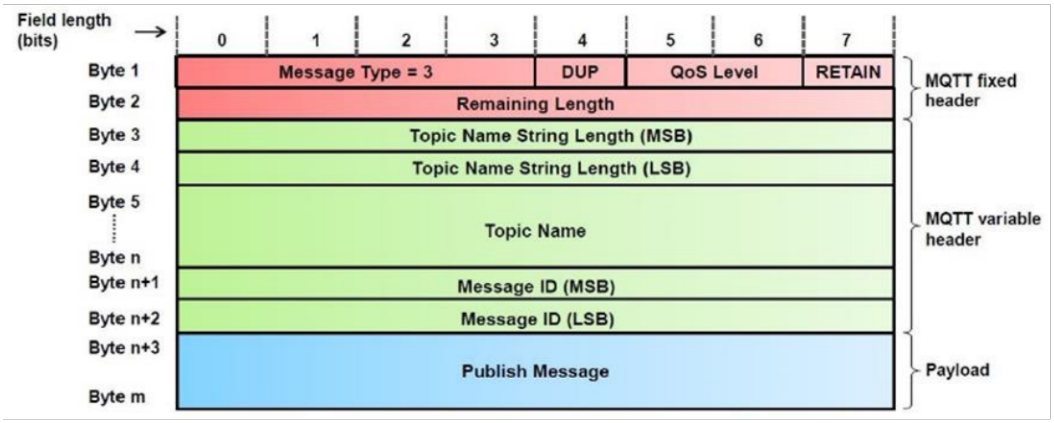


Figura 27: Mensaje PUBLISH MQTT [14].

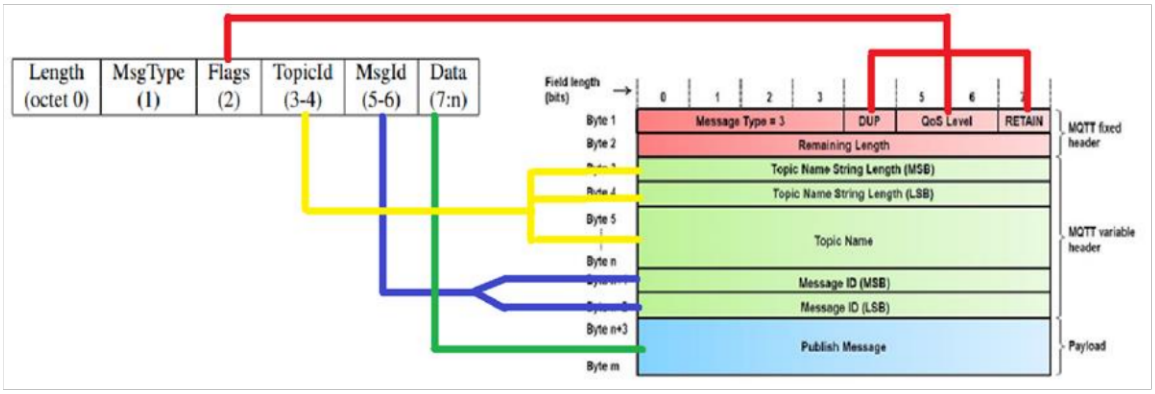


Figura 28: Esquema de correspondencia de campos.

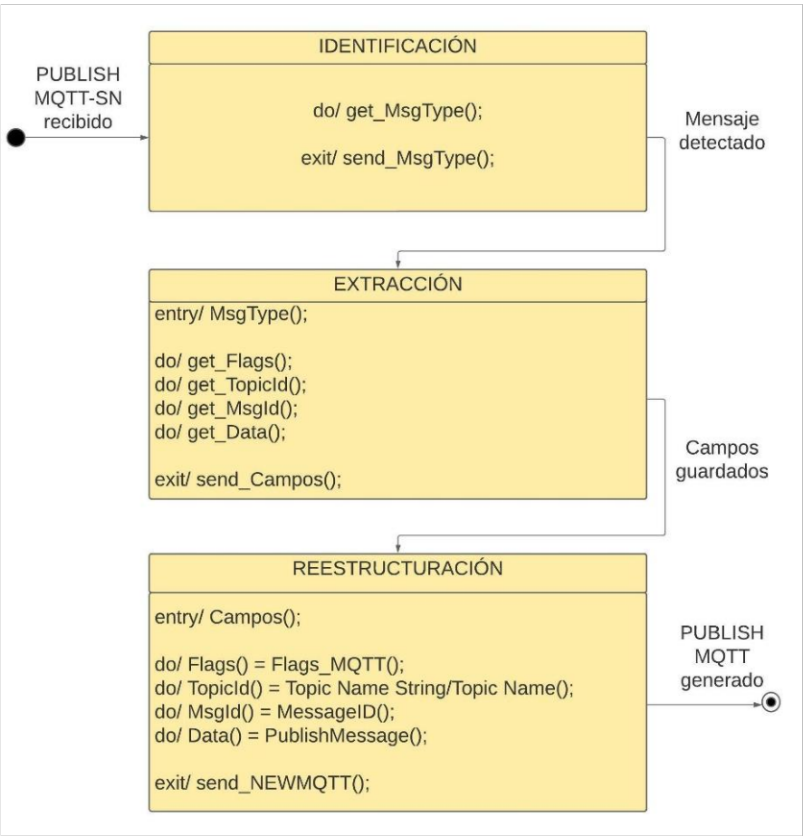


Figura 29: Proceso de conversión MQTT-SN a MQTT.

Usando la Figura 28 como referencia de la correspondencia de campos en la conversión de protocolos del mensaje PUBLISH se analizan los procesos realizados en el diagrama de estados, como se muestra en la Figura 29.

En el estado IDENTIFICACIÓN se analiza el campo MsgType con el identificador del mensaje PUBLISH (0X0C) y se envía en una acción de salida hacia el siguiente estado.

En el estado EXTRACCIÓN se realiza la adquisición de los campos Flags, TopicId, MsgId y Data, se almacenan en la variable “Campos()”, la cual es enviada al siguiente estado.

En el estado REESTRUCTURACIÓN se ingresa la variable con la información de los campos necesarios para crear el mensaje PUBLISH MQTT que posteriormente será enviado al bróker. En este estado se realizan las siguientes asignaciones:

Los valores de Length y MsgType son generados en el proceso de conversión dado que no se mantienen iguales entre los protocolos.

- Flags:

En la bandera DUP se realiza la asignación del mismo valor en el campo DUP del mensaje MQTT, en ambos protocolos indica si un mensaje es enviado por primera vez o no.

En la bandera QoS, se realiza la asignación del mismo valor en el campo QoS del mensaje MQTT, en ambos protocolos indica el nivel de calidad de servicio del mensaje PUBLISH.

En la bandera Retain se realiza la asignación del mismo valor en el campo RETAIN del mensaje MQTT, en ambos protocolos indica si se debe guardar el mensaje publicado.

Las banderas Will y CleanSession no se usan en este caso por lo que no son tomadas en

cuenta para la asignación.

- TopicId:

Contiene el identificador de topic asignado por el Gateway previamente en el estado REGISTRO. En el proceso de asignación se realiza una asociación del TopicId del nodo sensor con el Topic Name registrado en el Gateway, este será utilizado para la estructura del mensaje PUBLISH MQTT.

- MsgId:

Este campo es asignado a Message ID del mensaje MQTT y es relevante solo si se utiliza QoS de nivel 2, caso contrario tiene el valor 0x0000.

- Data:

Contiene el mensaje a publicar, es asignado al campo Publish Message del mensaje MQTT.

Después de realizar todas las asignaciones correspondientes, se envía el mensaje MQTT, que dentro del estado se denomina NEWMQTT. Se da por finalizado el proceso y se realiza la transición al estado DISPONIBLE como se observa en el diagrama de estados desarrollado.

- PUBREL

Length (octet 0)	MsgType (1)	MsgId (2-3)

Figura 30: Mensaje PUBREL MQTT-SN [2].

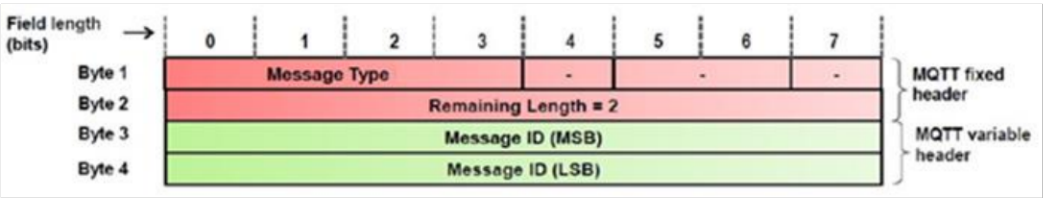


Figura 31: Mensaje PUBREL MQTT [14].

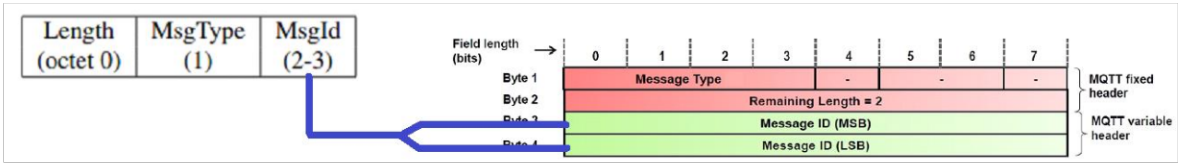


Figura 32: Esquema de correspondencia de campos.

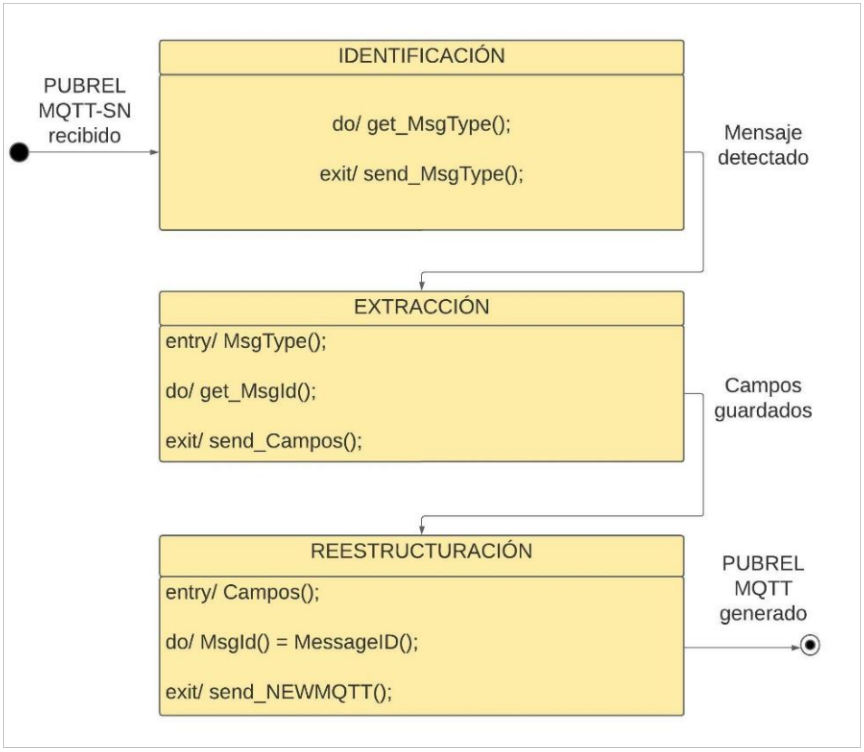


Figura 33: Proceso de conversión MQTT-SN a MQTT.

Usando la Figura 32 como referencia de la correspondencia de campos en la conversión de protocolos del mensaje PUBREL, se analizan los procesos realizados en el diagrama de estados, como se muestra en la Figura 33.

En el estado IDENTIFICACIÓN se analiza el campo MsgType con el identificador del mensaje PUBREL (0X10) y se envía en una acción de salida hacia el siguiente estado.

En el estado EXTRACCIÓN se realiza la adquisición del campo MsgId, se almacena en la variable “Campos()” y se envía al siguiente estado.

En el estado REESTRUCTURACIÓN se ingresa la variable con la información de los campos necesarios para crear el mensaje PUBREL MQTT que posteriormente será enviado al bróker. En este estado se realiza la siguiente asignación:

Los valores de Length y MsgType son generados en el proceso de conversión porque no se mantienen iguales entre los protocolos.

- MsgId:

Este campo es asignado a Message ID, tiene el mismo valor que en el mensaje PUBLISH que fue enviado previamente ya que es parte del flujo de mensajes de control de QoS nivel 2.

Después de realizar la asignación se envía el mensaje MQTT, el cual dentro del estado se denomina NEWMQTT. Se da por finalizado el proceso y se realiza la transición al estado DISPONIBLE como se observa en el diagrama de estados desarrollado.

- SUBSCRIBE

Length (octet 0)	MsgType (1)	Flags (2)	MsgId (3-4)	TopicName or TopicId (5:n) or (5-6)

Figura 34: Mensaje SUBSCRIBE MQTT-SN [2].

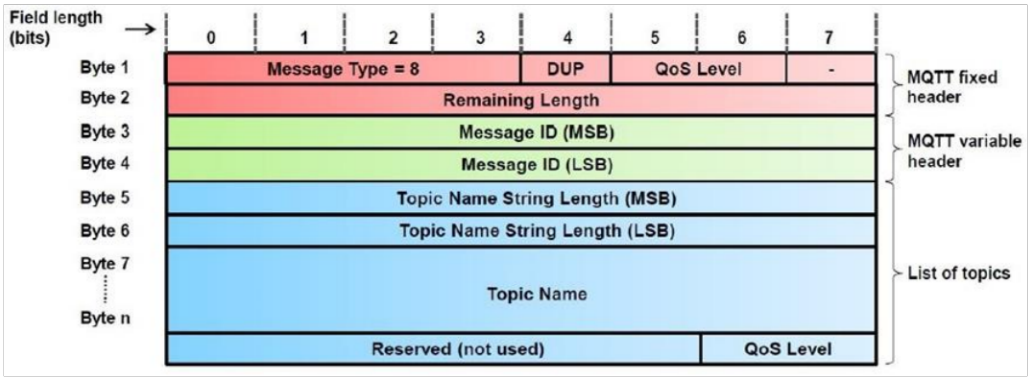


Figura 35: Mensaje SUBSCRIBE MQTT [14].

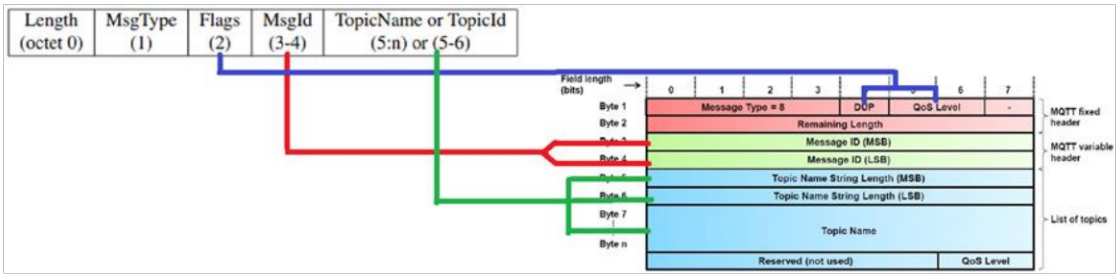


Figura 36: Esquema de correspondencia de campos.

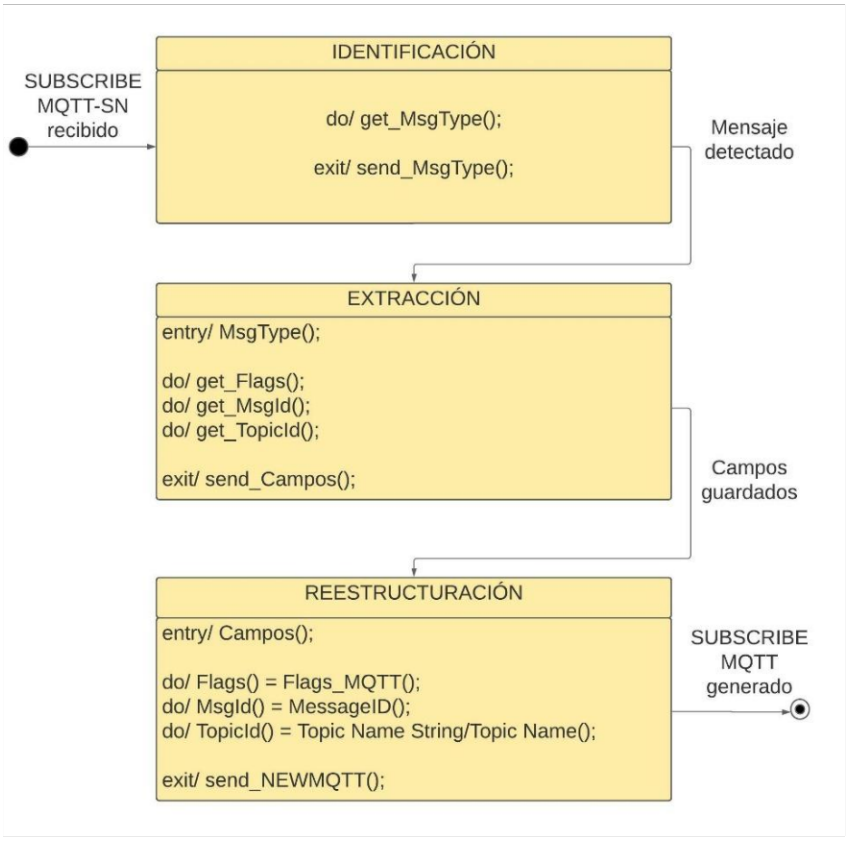


Figura 37: Proceso de conversión MQTT-SN a MQTT.

Usando la Figura 36 como referencia de la correspondencia de campos en la conversión

de protocolos del mensaje SUBSCRIBE se analizan los procesos realizados en el diagrama de estados, como se muestra en la Figura 37.

En el estado IDENTIFICACIÓN se analiza el campo MsgType con el identificador del mensaje SUBSCRIBE (0X12) y se envía en una acción de salida hacia el siguiente estado.

En el estado EXTRACCIÓN se realiza la adquisición de los campos Flags, MsgId y TopicId, se almacenan en la variable “Campos()” que es enviada al siguiente estado.

En el estado REESTRUCTURACIÓN se ingresa la variable con la información de los campos necesarios para crear el mensaje SUBSCRIBE MQTT, el cual posteriormente será enviado al bróker. En este estado se realizan las siguientes asignaciones:

Los valores de Length y MsgType son generados en el proceso de conversión dado que no se mantienen iguales entre los protocolos.

- Flags:

En la bandera DUP se realiza la asignación del mismo valor en el campo DUP del mensaje MQTT, en ambos protocolos indica si un mensaje es enviado por primera vez o no.

En la bandera QoS, se realiza la asignación del mismo valor en el campo QoS del mensaje MQTT, en ambos protocolos indica el nivel de calidad de servicio del mensaje SUBSCRIBE.

Las banderas Retain, Will y CleanSession no se usan en este caso por lo que no son tomadas en cuenta para la asignación.

- MsgId:

Este campo es asignado a Message ID del mensaje MQTT, y es usado para identificar el mensaje SUBACK que se enviará como respuesta a este mensaje.

- TopicId:

Contiene el identificador de topic asignado por el Gateway previamente en el estado REGISTRO. En el proceso de asignación se realiza una asociación del TopicId del nodo sensor con el Topic Name registrado en el Gateway, el cual será utilizado para la estructura del mensaje SUBSCRIBE MQTT.

Después de realizar todas las asignaciones correspondientes se envía el mensaje MQTT, que dentro del estado se denomina NEWMQTT. Se da por finalizado el proceso y se realiza la transición al estado DISPONIBLE como se observa en el diagrama de estados desarrollado.

- UNSUBSCRIBE

Length (octet 0)	MsgType (1)	Flags (2)	MsgId (3-4)	TopicName or TopicId (5:n) or (5-6)

Figura 38: Mensaje UNSUBSCRIBE MQTT-SN [2].

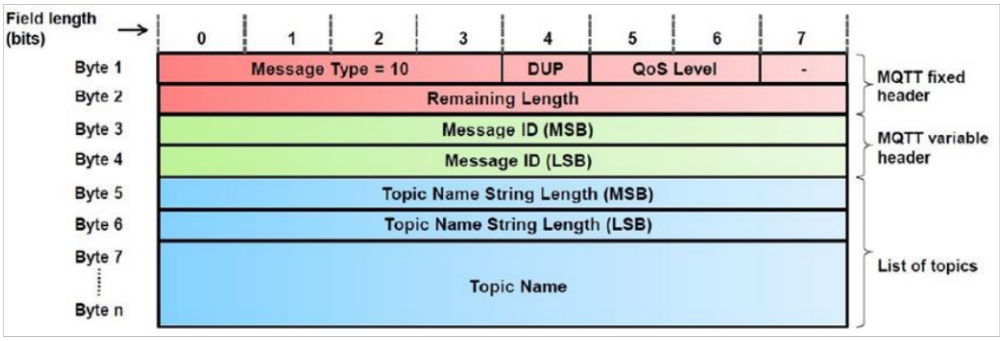


Figura 39: Mensaje UNSUBSCRIBE MQTT [14].

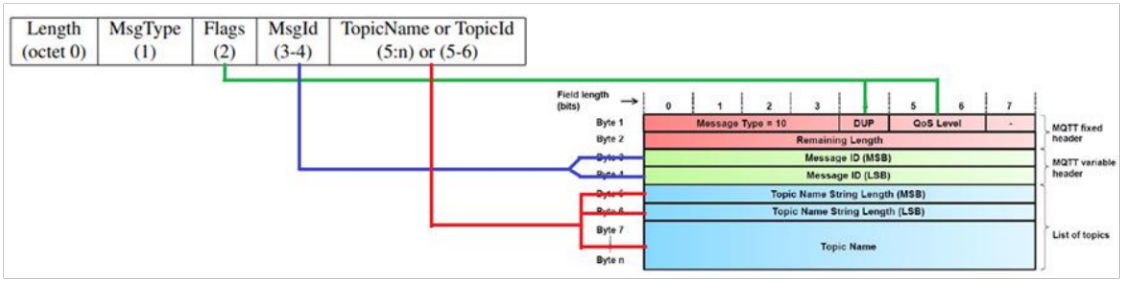


Figura 40: Esquema de correspondencia de campos.

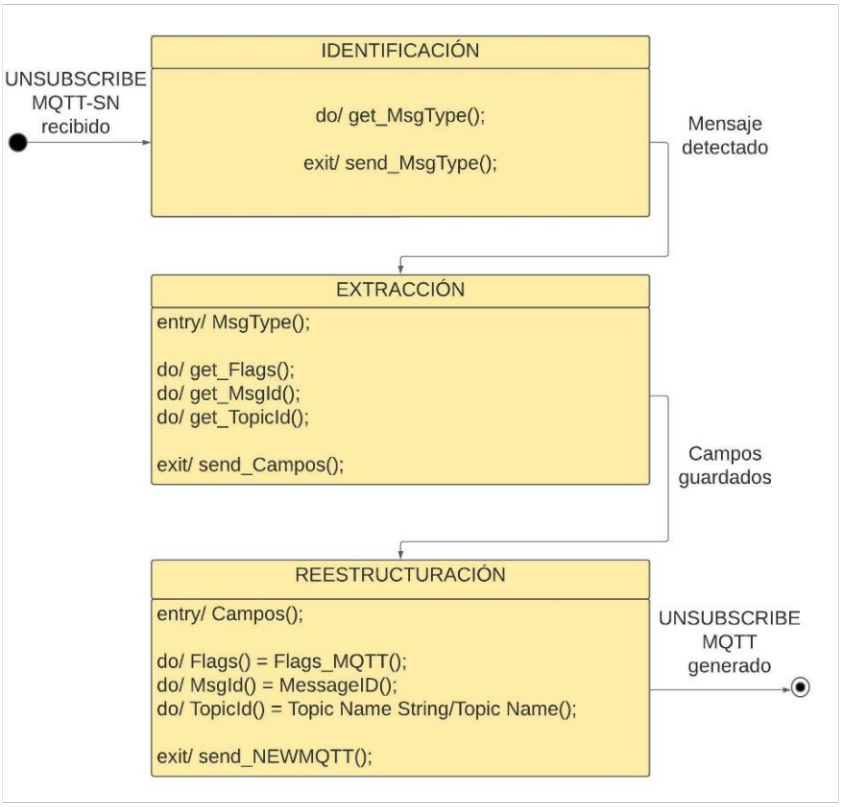


Figura 41: Proceso de conversión MQTT-SN a MQTT.

Usando la Figura 40 como referencia de la correspondencia de campos en la conversión de protocolos del mensaje UNSUBSCRIBE, se analizan los procesos realizados en el diagrama de estados, como se muestra en la Figura 41.

En el estado IDENTIFICACIÓN se analiza el campo MsgType con el identificador del mensaje UNSUBSCRIBE (0X14) y se envía en una acción de salida hacia el siguiente estado.

En el estado EXTRACCIÓN se realiza la adquisición de los campos Flags, MsgId y TopicId, se almacenan en la variable “Campos()” que es enviada al siguiente estado.

En el estado REESTRUCTURACIÓN se ingresa la variable con la información de los campos necesarios para crear el mensaje UNSUBSCRIBE MQTT que posteriormente será enviado al bróker. En este estado se realizan las siguientes asignaciones:

Los valores de Length y MsgType son generados en el proceso de conversión dado que no se mantienen iguales entre los protocolos.

- Flags:

En la bandera DUP se realiza la asignación del mismo valor en el campo DUP del mensaje MQTT, en ambos protocolos indica si un mensaje es enviado por primera vez o no.

En la bandera QoS, se realiza la asignación del mismo valor en el campo QoS del mensaje MQTT, en ambos protocolos indica el nivel de calidad de servicio del mensaje UNSUBSCRIBE.

Las banderas Retain, Will y CleanSession no se usan en este caso por lo que no son tomadas en cuenta para la asignación.

- MsgId:

Este campo es asignado a Message ID del mensaje MQTT y es usado para identificar el mensaje UNSUBACK que se enviará como respuesta a este mensaje.

- TopicId:

Contiene el identificador de topic asignado por el Gateway previamente en el estado REGISTRO. En el proceso de asignación se realiza una asociación del TopicId del nodo sensor con el Topic Name registrado en el Gateway, el cual será utilizado para la estructura del mensaje UNSUBSCRIBE MQTT.

Después de realizar todas las asignaciones correspondientes se envía el mensaje MQTT, que dentro del estado se denomina NEWMQTT. Se da por finalizado el proceso y se realiza la transición al estado DISPONIBLE como se observa en el diagrama de estados desarrollado.

- PINGREQ

Length (octet 0)	MsgType (1)	ClientId (optional) (2:n)

Figura 42: Mensaje PINGREQ MQTT-SN [2].

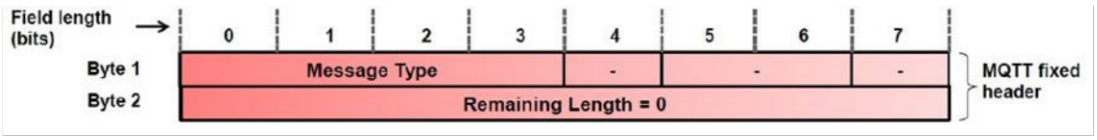


Figura 43: Mensaje PINGREQ MQTT [14].

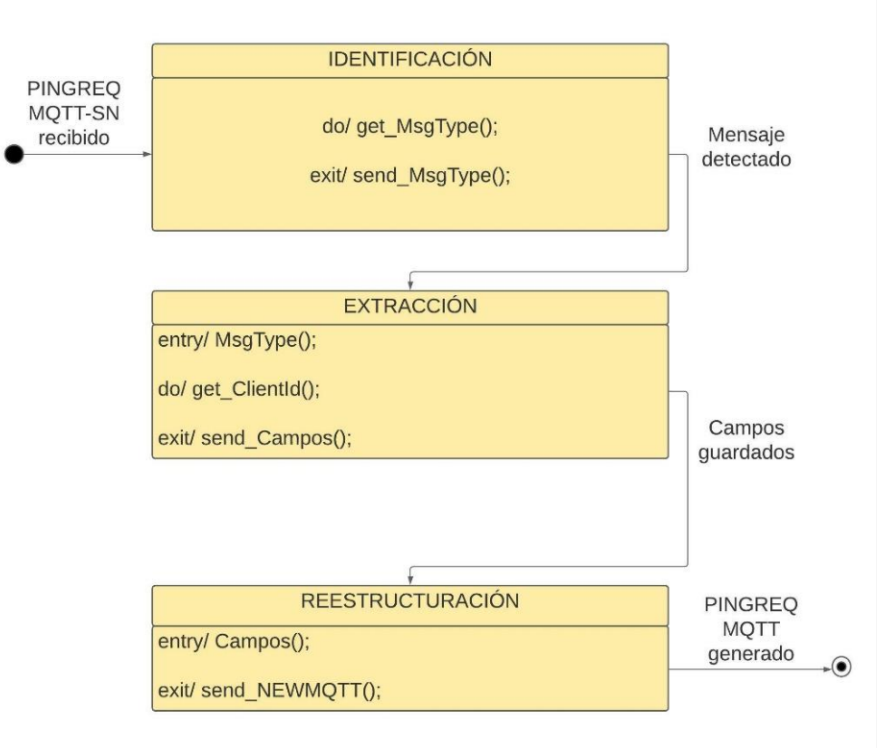


Figura 44: Proceso de conversión MQTT-SN a MQTT.

Como se puede observar en la Figura 42, la estructura del mensaje PINGREQ MQTT-SN no tiene campos que puedan entrar en el proceso de asignación definido en los estados de la Figura 44. Esto se debe a que el campo ClientId es opcional y no tiene un espacio asignado en la estructura del mensaje PINGREQ MQTT como se puede apreciar en la Figura 43.

El campo de ClientId es de uso específico entre el nodo sensor y el Gateway. Este es enviado desde el nodo sensor hacia el Gateway para notificar que cambió su estado de “sleeping” (reposo) a "awake" (activo) y está esperando mensajes.

Los valores de Length y MsgType son generados en el proceso de conversión dado que no se mantienen iguales entre los protocolos, como se explicó anteriormente el primer campo contiene el total de bytes del mensaje y el segundo campo se generará según los valores de la Tabla 1.1.

El mensaje MQTT generado es producto de los procesos asociados a los estados, pero no se realiza ninguna asignación de campos correspondientes como en los casos anteriores. Después de generar el mensaje, se envía y se da por finalizado el proceso, como consecuencia se realiza la transición al estado DISPONIBLE.

- DISCONNECT

Length (octet 0)	MsgType (1)	Duration (optional) (2-3)

Figura 45: Mensaje DISCONNECT MQTT-SN [2].

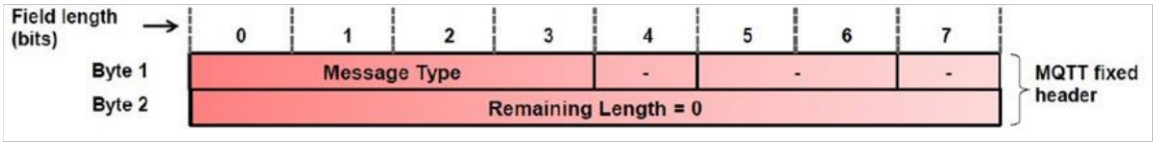


Figura 46: Mensaje DISCONNECT MQTT [14].

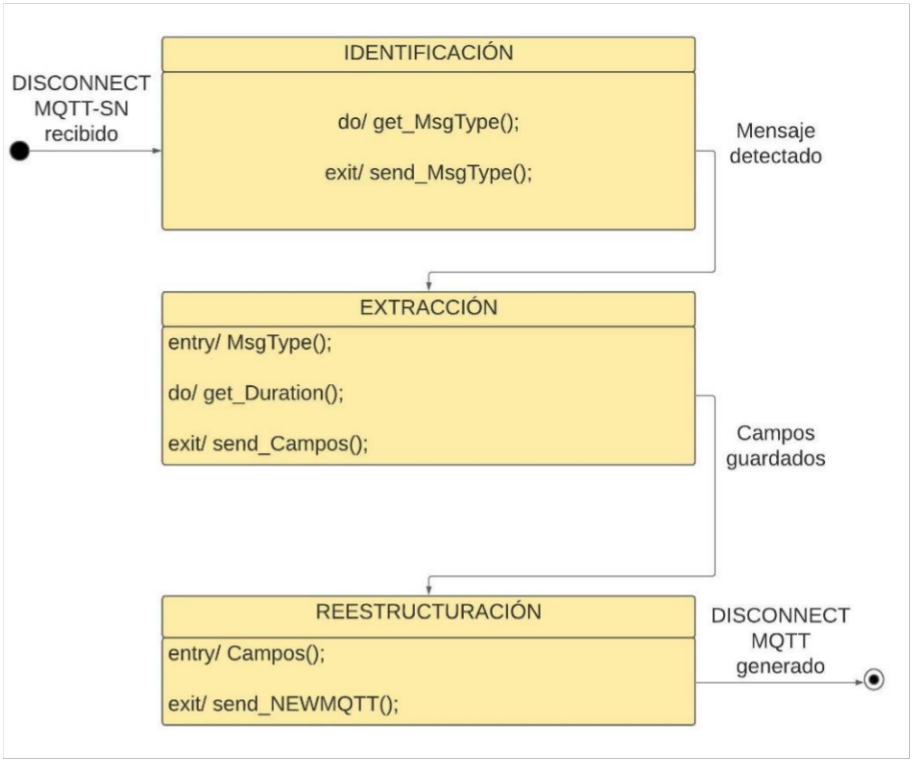


Figura 47: Proceso de conversión MQTT-SN a MQTT.

Como se puede observar en la Figura 45, la estructura del mensaje DISCONNECT MQTT-SN no tiene campos que puedan entrar en el proceso de asignación definido en los estados de la Figura 47. Esto se debe a que el campo Duration es opcional y no tiene un espacio asignado en la estructura del mensaje DISCONNECT MQTT, tal como se puede apreciar en la Figura 46.

El campo Duration está dirigido al Gateway debido a que se envía de parte del cliente para notificar que pasa a estado de reposo o “asleep”.

Los valores de Length y MsgType son generados en el proceso de conversión dado que no se mantienen iguales entre los protocolos.

El mensaje MQTT generado es producto de los procesos asociados a los estados, pero no se realiza ninguna asignación de campos correspondientes como en los casos anteriores. Después de generar el mensaje, se envía y se da por finalizado el proceso, como resultado se realiza la transición al estado DISPONIBLE.

2.2 Conversión de protocolo MQTT a MQTT-SN

En la anterior sección se analizaron los mensajes que requieren una conversión de protocolo MQTT-SN a MQTT para poder ser enviados al bróker. En el protocolo MQTT-SN también se definen mensajes que viajan en el otro sentido, por ende necesitan una conversión de protocolo MQTT a MQTT-SN en el Gateway para ser procesados por el cliente. A continuación, se presentan los mensajes que forman parte del protocolo MQTT-SN y su proceso de conversión previo.

En la Figura 48 se muestra el diagrama de estados diseñado en este trabajo para representar el funcionamiento del convertidor de protocolos MQTT a MQTT-SN, del cual se profundizará su análisis a continuación.

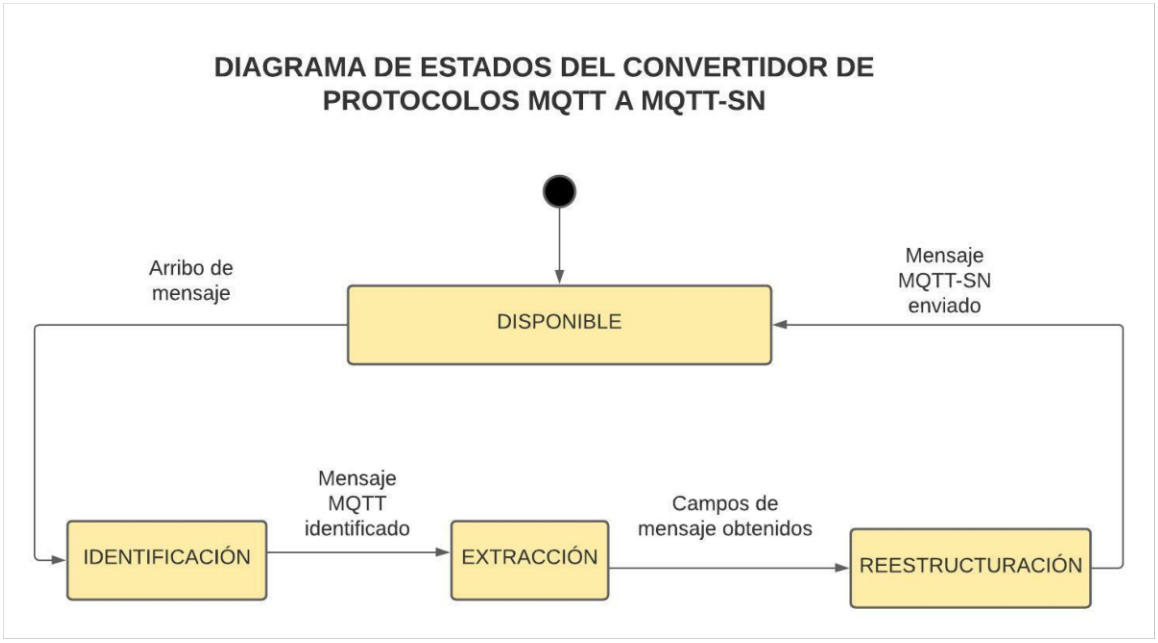


Figura 48: Diagrama de estados del convertidor de protocolos MQTT a MQTT-SN.

Para la elaboración del diagrama de la Figura 48 se crearon diferentes estados, cada uno implica uno o más procesos que contribuyen en la funcionalidad del convertidor de protocolos por lo que se procede a explicar cada uno.

DISPONIBLE

Es el estado principal del diagrama en el cual el Gateway acepta mensajes desde el bróker para procesarlos y enviarlos al nodo cliente. Si se presenta un evento de arribo de mensaje se procede a realizar una transición al estado de IDENTIFICACIÓN. En la comunicación bróker - Gateway ya no se requiere una conexión previa entre ambos componentes por lo que se suprimen los estados de RECEPCIÓN, REGISTRO y ESTABLECIMIENTO.

IDENTIFICACIÓN

En este estado se define el tipo de mensaje enviado por el bróker. Este proceso se realiza mediante el análisis de la cabecera del mensaje recibido, específicamente el campo Message Type, el cual contiene el número de identificación que corresponde a los

mensajes definidos en el protocolo MQTT. Después de analizar el campo Message Type se define el mensaje recibido y se procede a realizar una transición al estado EXTRACCIÓN.

EXTRACCIÓN

Después del proceso de identificación del mensaje MQTT se analiza la estructura del mensaje ingresado y dependiendo del mismo, se tomarán ciertos campos que serán usados para la creación del mensaje MQTT-SN en el estado siguiente. Una vez terminado el proceso de adquisición de campos del mensaje se realiza una transición al estado REESTRUCTURACIÓN.

REESTRUCTURACIÓN

En este estado se reciben los campos del mensaje MQTT para su conversión hacia el protocolo MQTT-SN y así generar un mensaje que pueda ser procesado en el destino. En REESTRUCTURACIÓN se realiza una asignación de un campo MQTT en el correspondiente campo MQTT-SN, para esto en algunos casos se requiere información previa del nodo sensor almacenada en el Gateway.

Después del proceso de generación del mensaje MQTT-SN ejecutado en el presente estado, se realiza el envío del mensaje hacia el nodo sensor y con este evento se produce una transición hacia el estado base DISPONIBLE.

A continuación, se presentan las estructuras del mensaje MQTT recibido y del mensaje MQTT-SN transferido, un esquema de la correspondencia de campos en la conversión de cada mensaje y los procesos realizados en los últimos 3 estados del diagrama desarrollado en este trabajo.

- CONNACK

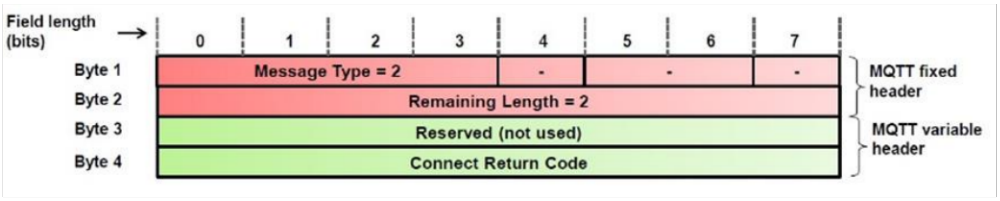


Figura 49: Mensaje CONNACK MQTT [14].

Length (octet 0)	MsgType (1)	ReturnCode (2)

Figura 50: Mensaje CONNACK MQTT-SN [2].

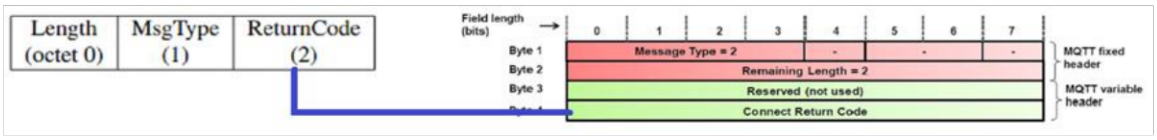


Figura 51: Esquema de correspondencia de campos.

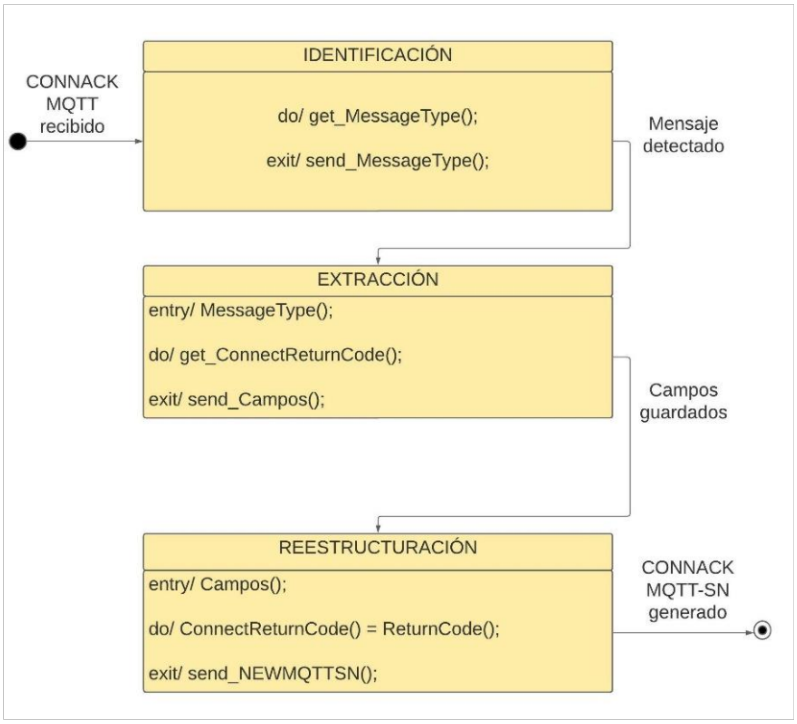


Figura 52: Proceso de conversión MQTT a MQTT-SN.

Usando la Figura 51 como referencia de la correspondencia de campos en la conversión de protocolos del mensaje CONNACK se analizan los procesos realizados en el diagrama de estados, como se muestra en la Figura 52.

En el estado IDENTIFICACIÓN se analiza el campo Message Type con el identificador del mensaje CONNACK (2) y se envía en una acción de salida hacia el siguiente estado.

En el estado EXTRACCIÓN se realiza la adquisición del campo Connect Return Code y se almacena en la variable “Campos()” la cual es enviada al siguiente estado.

En el estado REESTRUCTURACIÓN se ingresa la variable con la información de los campos necesarios para crear el mensaje CONNACK MQTT-SN que posteriormente será enviado al nodo cliente. En este estado se realiza la siguiente asignación:

Los valores de Length y MsgType son generados en el proceso de conversión dado que no se mantienen iguales entre los protocolos.

- Connect Return Code:

Este campo indica si la conexión fue aceptada (0x00), rechazada debido a una versión de protocolo incompatible (0x01), rechazada debido a un identificador de cliente incorrecto (0x02), rechazada debido a que el servidor no está disponible (0x03), rechazada debido a un nombre de usuario/contraseña incorrectos (0x04) o rechazada debido a que es una conexión no autorizada (0x05).

En el Gateway se realiza una correlación entre los valores de Connect Return Code anteriormente mencionados y los valores definidos en el protocolo MQTT-SN para el campo Return Code. En el campo del mensaje MQTT-SN se define si la conexión fue aceptada (0x00), rechazada por congestión (0x01), rechazada debido a un Topic Id invalido (0x02) o

rechazada porque no soporta la versión del protocolo.

Después de realizar la asignación se envía el mensaje MQTT-SN, que dentro del estado se denomina NEWMQTTSN, se da por finalizado el proceso y se realiza la transición al estado DISPONIBLE como se observa en el diagrama de estados desarrollado.

- PUBACK

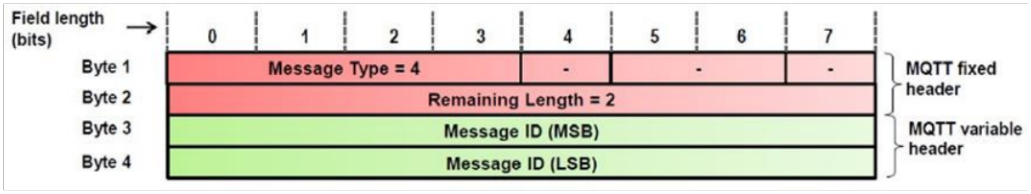


Figura 53: Mensaje PUBACK MQTT [14].

Length (octet 0)	MsgType (1)	TopicId (2,3)	MsgId (4,5)	ReturnCode (6)

Figura 54: Mensaje PUBACK MQTT-SN [2].

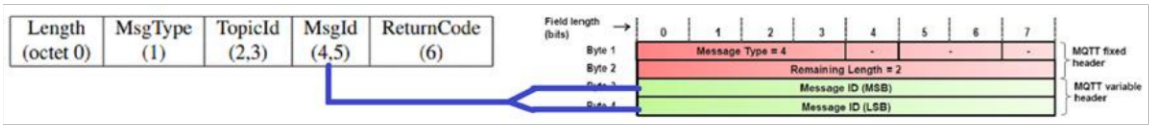


Figura 55: Esquema de correspondencia de campos.

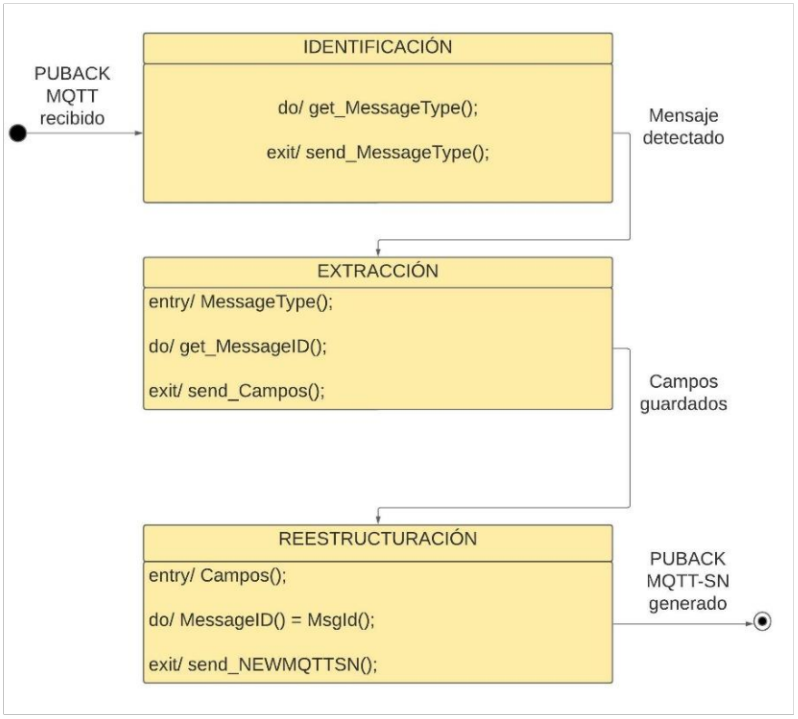


Figura 56: Proceso de conversión MQTT a MQTT-SN.

Usando la Figura 55 como referencia de la correspondencia de campos en la conversión de protocolos del mensaje PUBACK se analizan los procesos realizados en el diagrama de estados, como se muestra en la Figura 56.

En el estado IDENTIFICACIÓN se analiza el campo Message Type con el identificador del

mensaje PUBACK (4) y se envía en una acción de salida hacia el siguiente estado.

En el estado EXTRACCIÓN se realiza la adquisición del campo Message ID y se almacena en la variable “Campos()”.

En el estado REESTRUCTURACIÓN se ingresa la variable con la información de los campos necesarios para crear el mensaje PUBACK MQTT-SN, el cual se enviará al nodo cliente. En este estado se realiza la siguiente asignación:

Los valores de Length y MsgType son generados en el proceso de conversión dado que no se mantienen iguales entre los protocolos.

- Message ID:

Este campo es asignado a MsgId del mensaje MQTT-SN y es usado para identificarlo como respuesta del mensaje PUBLISH enviado previamente por el cliente.

El valor del campo ReturnCode es ingresado por el Gateway ya que no es generado por el bróker, este campo indicara si la acción de publicar fue aceptada o rechazada.

El campo TopicId contiene el identificador de topic asignado por el Gateway al cliente.

Después de realizar la asignación se envía el mensaje MQTT-SN, que dentro del estado se denomina NEWMQTTSN y se realiza la transición al estado DISPONIBLE.

- PUBREC y PUBCOMP

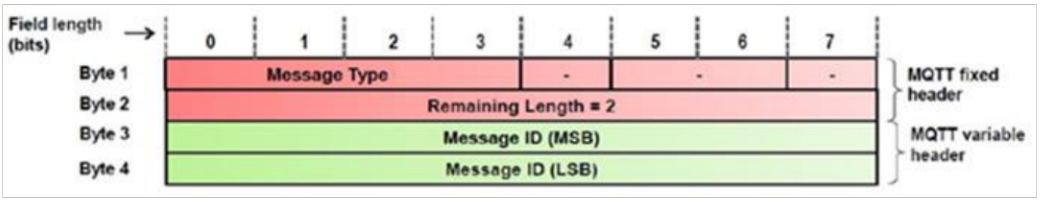


Figura 57: Mensaje PUBREC y PUBCOMP MQTT [14].

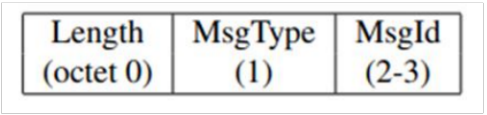


Figura 58: Mensaje PUBREC y PUBCOMP MQTT-SN [2].

Figura 59: Esquema de correspondencia de campos.

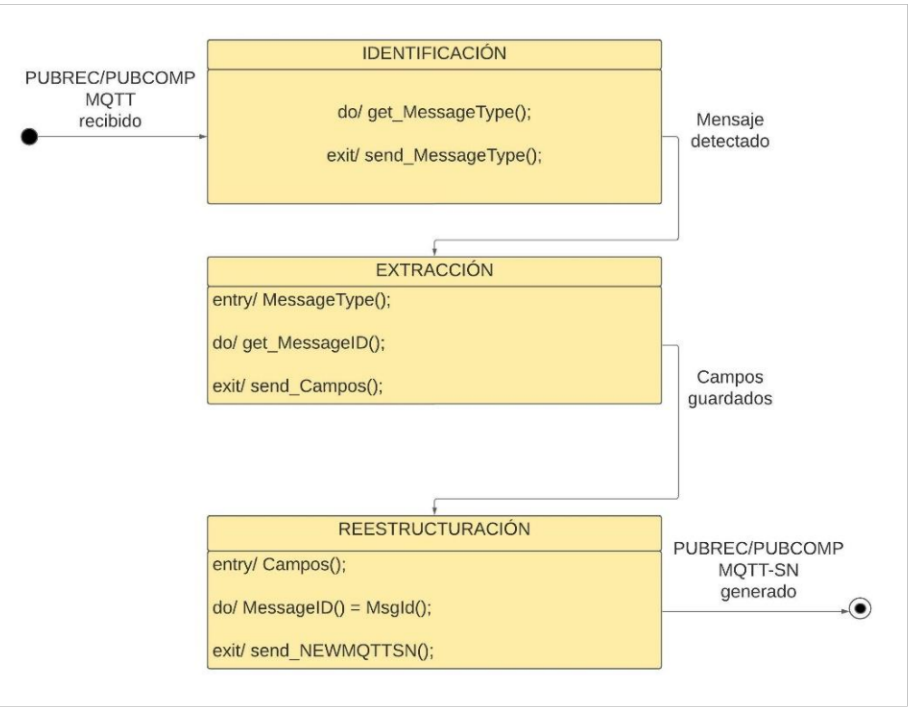


Figura 60: Proceso de conversión MQTT a MQTT-SN.

Usando la Figura 59 como referencia de la correspondencia de campos en la conversión de protocolos de los mensajes PUBREC/PUBCOMP se analizan los procesos realizados en el diagrama de estados, como se muestra en la Figura 60.

En el estado IDENTIFICACIÓN se analiza el campo Message Type con el identificador del mensaje PUBREC (5) o PUBCOMP (7) y se envía en una acción de salida hacia el siguiente estado.

En el estado EXTRACCIÓN se realiza la adquisición del campo Message ID y se almacena en la variable “Campos()”.

En el estado REESTRUCTURACIÓN se ingresa la variable con la información necesaria para crear el mensaje PUBREC o PUBCOMP MQTT-SN que posteriormente será enviado. En este estado se realiza la siguiente asignación:

Los valores de Length y MsgType son generados en el proceso de conversión dado que no se mantienen iguales entre los protocolos.

- Message ID:

Este campo es asignado a MsgId del mensaje MQTT-SN y debe ser igual que el enviado en el mensaje PUBLISH enviado previamente por el cliente.

Después de realizar la asignación se envía el mensaje MQTT-SN, que dentro del estado se denomina NEWMQTTSN. Después se realiza la transición al estado DISPONIBLE como se observa en el diagrama de estados desarrollado.

- SUBACK

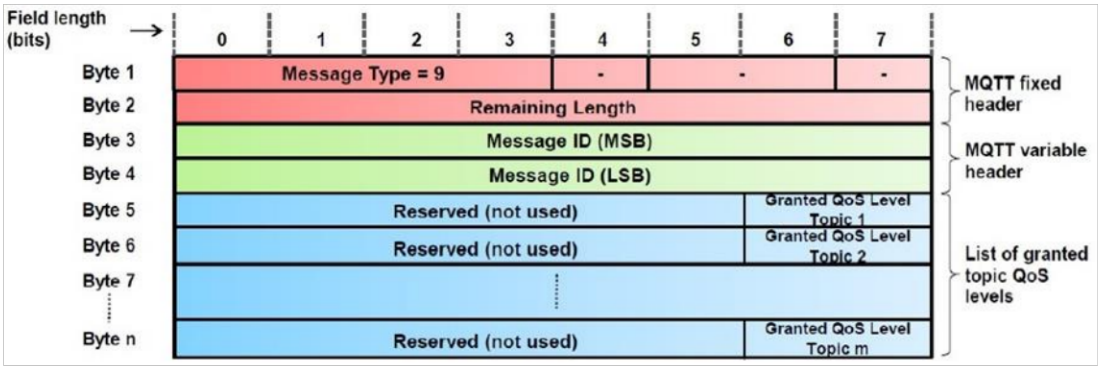


Figura 61: Mensaje SUBACK MQTT [14].

Length (octet 0)	MsgType (1)	Flags (2)	TopicId (3,4)	MsgId (5,6)	ReturnCode (7)

Figura 62: Mensaje SUBACK MQTT-SN [2].

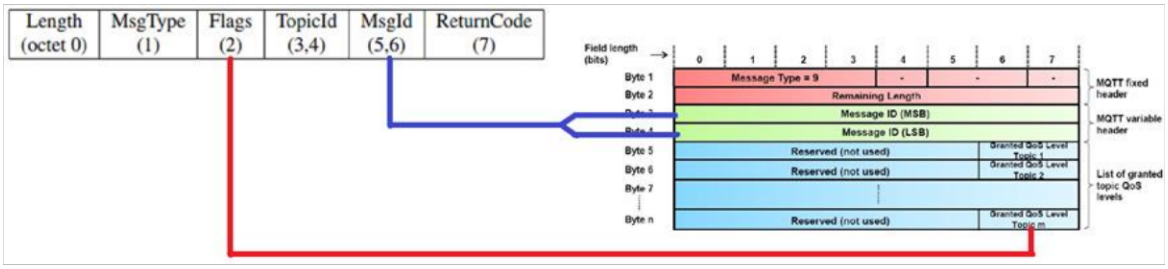


Figura 63: Esquema de correspondencia de campos.

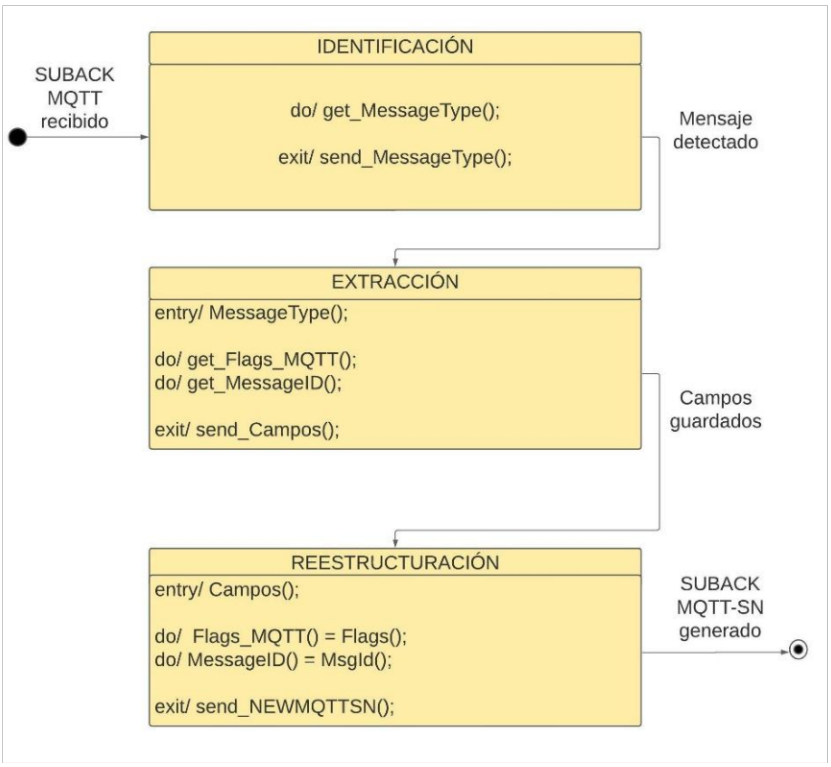


Figura 64: Proceso de conversión MQTT a MQTT-SN.

Usando la Figura 63 como referencia de la correspondencia de campos en la conversión de protocolos del mensaje SUBACK se analizan los procesos realizados en el diagrama de

estados, como se muestra en la Figura 64.

En el estado IDENTIFICACIÓN se analiza el campo Message Type con el identificador del mensaje SUBACK (9) y se envía en una acción de salida hacia el siguiente estado.

En el estado EXTRACCIÓN se realiza la adquisición del campo Flags y Message ID para almacenarlos en la variable “Campos()”, ésta es enviada al siguiente estado.

En el estado REESTRUCTURACIÓN se ingresa la variable con la información de los campos necesarios para crear el mensaje SUBACK MQTT-SN que posteriormente será enviado al nodo cliente. En este estado se realiza la siguiente asignación:

Los valores de Length y MsgType son generados en el proceso de conversión dado que no se mantienen iguales entre los protocolos.

- Flags_MQTT:

Se toma el valor de uno de los tres indicadores del nivel de QoS presentes en el mensaje y se asignan al apartado de QoS del campo Flags en el mensaje MQTT-SN.

- Message ID:

Este campo es asignado a MsgId del mensaje MQTT-SN y debe ser igual que el enviado en el mensaje SUBSCRIBE enviado previamente por el cliente.

El valor del campo ReturnCode es ingresado por el Gateway ya que no es generado por el bróker, este campo indicará si la acción de publicar fue aceptada o rechazada.

El campo TopicId contiene el identificador de topic asignado por el Gateway al cliente.

Después de realizar la asignación se envía el mensaje MQTT-SN, que dentro del estado se denomina NEWMQTTSN, se da por finalizado el proceso y se realiza la transición al estado DISPONIBLE.

- UNSUBACK

Figura 65: Mensaje UNSUBACK MQTT [14].

Length (octet 0)	MsgType (1)	MsgId (2-3)

Figura 66: Mensaje UNSUBACK MQTT-SN [2].

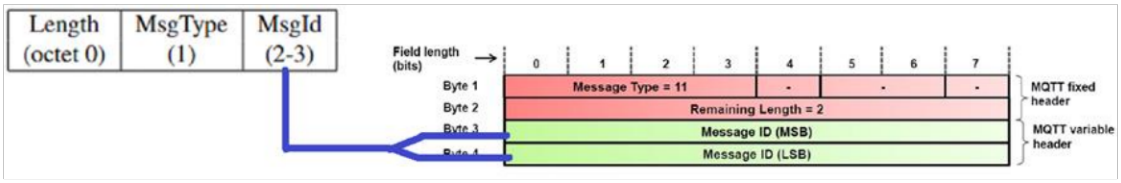


Figura 67: Esquema de correspondencia de campos.

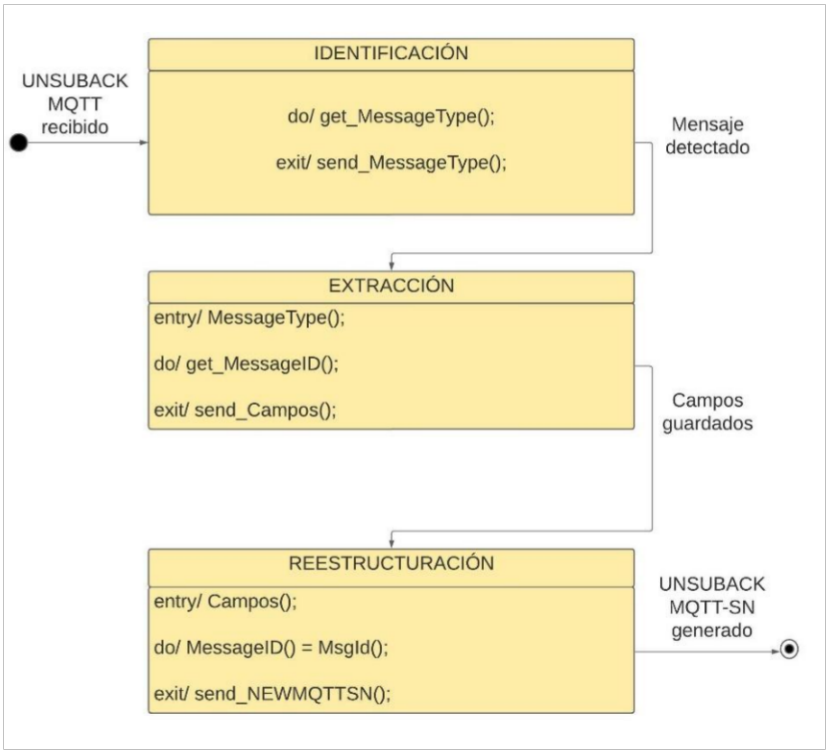


Figura 68: Proceso de conversión MQTT a MQTT-SN.

Usando la Figura 67 como referencia de la correspondencia de campos en la conversión de protocolos del mensaje UNSUBACK se analizan los procesos realizados en el diagrama de estados, como se muestra en la Figura 68.

En el estado IDENTIFICACIÓN se analiza el campo Message Type con el identificador del mensaje UNSUBACK (11) y se envía en una acción de salida hacia el siguiente estado.

En el estado EXTRACCIÓN se realiza la adquisición del campo Message ID y se almacena en la variable “Campos()”.

En el estado REESTRUCTURACIÓN se ingresa la variable con la información necesaria para crear el mensaje UNSUBACK MQTT-SN que será enviado. En este estado se realiza la siguiente asignación:

Los valores de Length y MsgType son generados en el proceso de conversión dado que no se mantienen iguales entre los protocolos.

- Message ID:

Este campo es asignado a MsgId del mensaje MQTT-SN y debe ser igual que el enviado en el mensaje UNSUBSCRIBE enviado previamente por el cliente.

Después de realizar la asignación se envía el mensaje MQTT-SN, que dentro del estado se denomina NEWMQTTSN, se da por finalizado el proceso y se realiza la transición al estado DISPONIBLE como se observa en el diagrama de estados desarrollado.

- PINGRESP

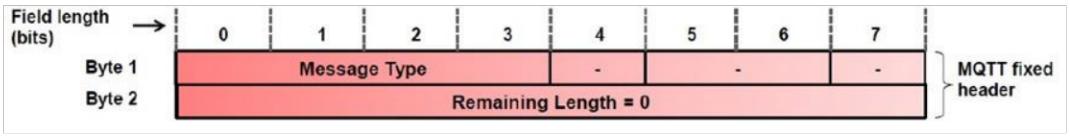


Figura 69: Mensaje PINGRESP MQTT [14].

Length (octet 0)	MsgType (1)

Figura 70: Mensaje PINGRESP MQTT-SN [2].

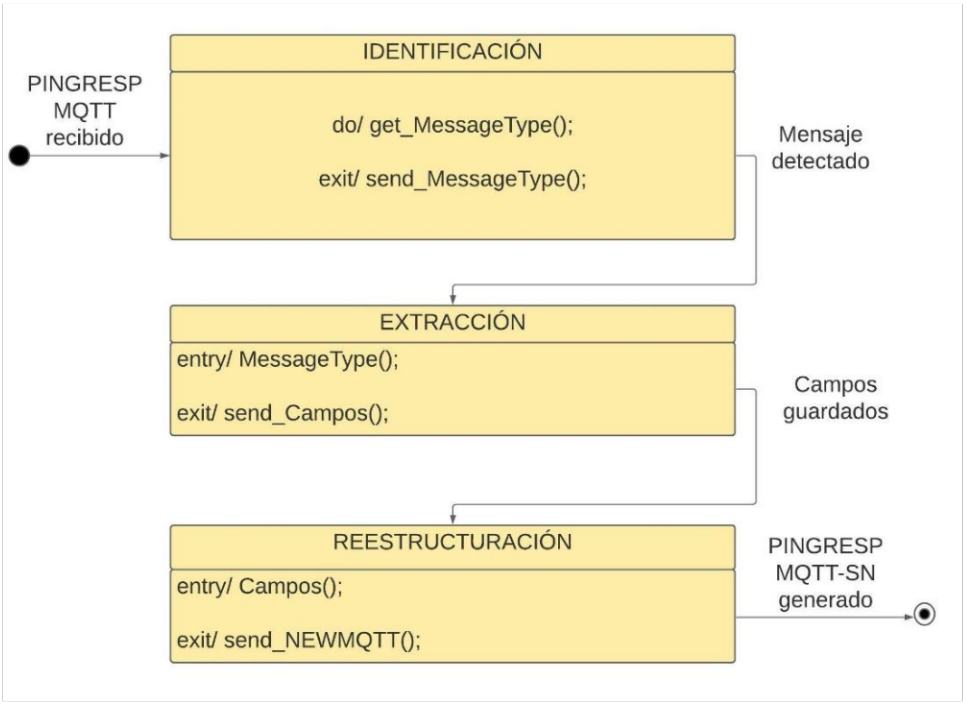


Figura 71: Proceso de conversión MQTT a MQTT-SN.

Como se puede observar en la Figura 69, la estructura del mensaje PINGRESP no tiene campos que puedan entrar en el proceso de asignación definido en los estados de la Figura 71.

Los valores de Length y MsgType son generados en el proceso de conversión dado que no se mantienen iguales entre los protocolos, como se explicó anteriormente el primer campo contiene el total de bytes del mensaje y el segundo campo se generará según los valores de la Tabla 1.2.

El mensaje MQTT-SN generado es producto de los procesos asociados a los estados, pero no se realiza ninguna asignación de campos correspondientes como en los casos anteriores. Después de generar el mensaje, se envía y se da por finalizado el proceso y

por ende se realiza la transición al estado DISPONIBLE.

2.3 Desarrollo de los diagramas de estado del convertidor de protocolos en UPPAAL Model Checker

Una vez analizados cada uno de los estados y procesos realizados en el diagrama diseñado en este trabajo, es clave probar su validez de operación para que el resultado final sirva de referencia para una futura implementación.

Como se explicó en el anterior capitulo, el software UPPAAL se utilizará en este trabajo para validar el comportamiento del diagrama de estados. A continuación, se presenta el desarrollo de los diagramas del convertidor de protocolos en la herramienta, indicando la lógica usada para las transiciones de estados y las condiciones de cada evento.

2.3.1 Convertidor de protocolo MQTT-SN a MQTT

Debido a que en UPPAL se valida la operación del sistema, se necesita empezar definiendo el comportamiento que tendrá un mensaje en el diagrama de estados del convertidor de protocolos MQTT-SN a MQTT.

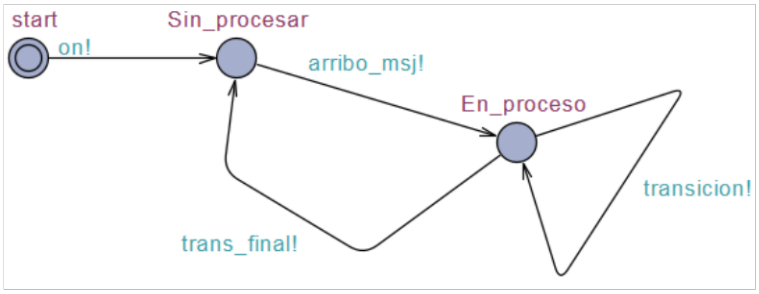


Figura 72: Diagrama de comportamiento de un mensaje desarrollado en UPPAAL.

En el diagrama de la Figura 72 se definen los estados “Sin_procesar” y “En_proceso” para representar el comportamiento de un mensaje. Para cambiar del estado inicial o “start” a “Sin_procesar” se define una transición de inicio “on!”, ésta tiene un signo de exclamación a su lado para que el software reconozca que es una sentencia obligatoria. Para representar el origen de un mensaje se creó el estado “Sin_procesar”, la transición “arribo_msj!” se utiliza para cambiar a “En_proceso”, ésta representa el evento de llegada del mensaje y da inicio al procesamiento que se realiza en el diagrama de estados.

El denominado “En_proceso” simboliza todos los estados diseñados en el diagrama, en los cuales se realizan los diferentes procesos sobre el mensaje para lograr la conversión de protocolos. En ese estado se define un bucle con la sentencia “transicion!”, la cual representa todas las transiciones de estados que sufrirá el mensaje dentro del diagrama. Este bucle seguirá hasta que el mensaje llega al último estado, en este se utilizará la sentencia “trans_final!” para definir la finalización de todo el procesamiento y la vuelta hacia el estado “Sin_procesar” para que se pueda validar el comportamiento del diagrama de

estados en el caso de otro arribo de mensaje.

Después de realizar el diagrama del comportamiento de un mensaje que entra al sistema, se procede a desarrollar el diagrama de estados del convertidor de protocolos MQTT-SN a MQTT presentado en la Figura 13.

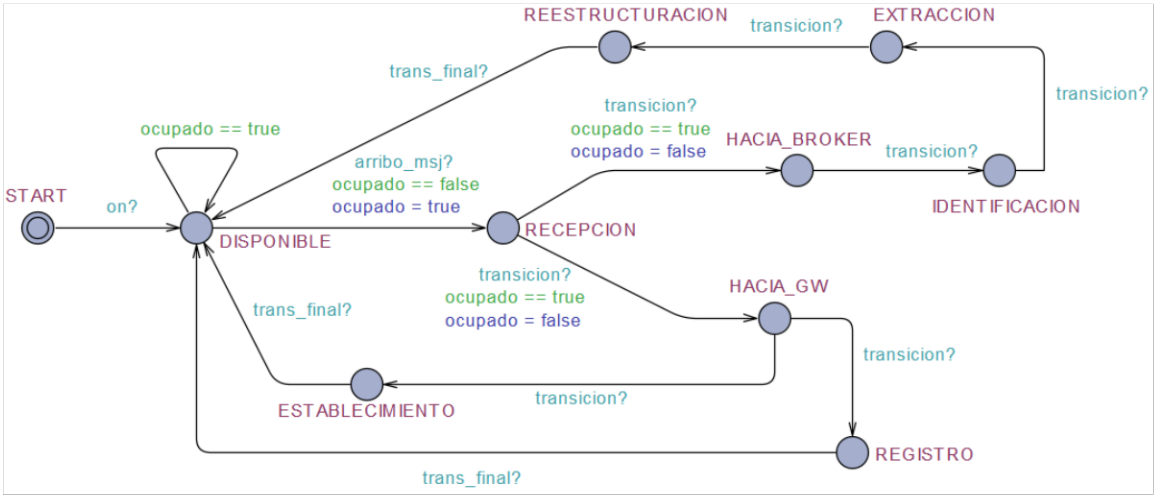


Figura 73: Diagrama de estados del convertidor de protocolos MQTT-SN a MQTT desarrollado en UPPAL.

En la Figura 73 se presenta el diagrama desarrollado, con los estados, eventos y transiciones ya analizados en la sección anterior. Para que se pueda realizar la validación del comportamiento del diagrama en UPPAAL, se utilizan las transiciones definidas en el diagrama de la Figura 72 pero acompañadas de un signo de pregunta (?) debido a que éste es el sistema a analizar y el software al simular el comportamiento del diagrama la considera una posible transición al encontrarse en un estado.

Para la simulación de las transiciones entre estados se definió la variable de tipo booleana “ocupado”, la cual controla que solo un mensaje pase al estado de “RECEPCION”, ya que cuando se produce la transición “arribo_msj” se realiza una actualización de la variable de falso a verdadero. Con ese cambio de variable, cualquier otro mensaje deberá esperar en “DISPONIBLE” hasta que se procese el mensaje y se realice una transición con actualización de valor de “ocupado”.

Una vez terminados los procesos sobre un mensaje se realiza una transición “trans_final” de vuelta hacia el estado “DISPONIBLE” para poder simular el arribo de otro mensaje y así validar la lógica de las transiciones del diagrama de estados diseñado.

2.3.2 Convertidor de protocolo MQTT a MQTT-SN

Al igual que en la sección anterior, se presenta a continuación el diagrama de estados del convertidor de protocolos MQTT a MQTT-SN desarrollado en UPPAAL.

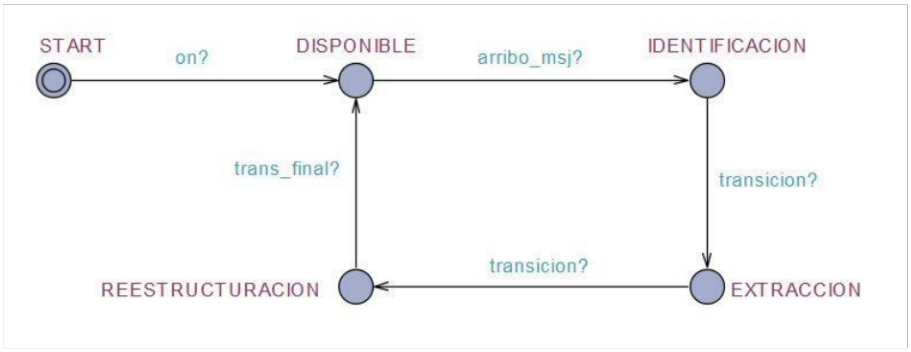


Figura 74: Diagrama de estados del convertidor de protocolos MQTT a MQTT-SN desarrollado en UPPAL.

El diagrama del comportamiento que tendrá un mensaje en el sistema es el presentado en la Figura 72, esto se debe a que se toman los mismos parámetros para verificar ambos diagramas de estados diseñados en el presente trabajo.

En este caso no se requiere el uso de la variable de tipo booleana debido a que en el sentido de conversión MQTT a MQTT-SN no se tienen mensajes de conexión primaria ni registro con el Gateway, tal como ocurre en la comunicación cliente – GW.

Tampoco se incluye el lazo de transición en el estado “DISPONIBLE” ya que en este caso cualquier mensaje tendrá el mismo procesamiento a penas ingrese al diagrama de estados.

3 RESULTADOS

Este capítulo se presentan los resultados de la validación del comportamiento de los diagramas diseñados previamente. Para facilitar el análisis se presentan capturas de pantalla del software UPPAAL, en el cual se realiza la verificación de cada diagrama de estados del convertidor de protocolos.

3.1 Validación del comportamiento del diagrama de estados del convertidor de protocolo MQTT-SN a MQTT

En esta sección se presenta la validación del diagrama de estados del convertidor de protocolo MQTT-SN a MQTT en UPPAAL. Con el fin de verificar tres escenarios a la vez, en el software se estructuró un sistema compuesto por tres diagramas de estados. En el primero (MQTTSN_ESTAB) se analiza el procesamiento sobre un mensaje de establecimiento de conexión primaria entre el nodo cliente y el nodo Gateway. En el segundo (MQTTSN_REGIST) se analiza el arribo de un mensaje de registro enviado desde el cliente al Gateway y en el tercero (MQTTSN_A_BROKERMQTT) se presenta el comportamiento del diagrama de estados cuando llega un mensaje del protocolo MQTT-SN que va dirigido hacia el bróker MQTT.

En el Anexo I se adjunta una captura de pantalla de la ventana de simulación completa del software UPPAAL, en la cual se puede visualizar los diagramas de los tres casos mencionados y las transiciones disponibles para ejecutar la verificación de cada diagrama.

Para iniciar el proceso de validación, en el software se efectuó la primera transición en los tres diagramas para que todos empiecen en el estado “DISPONIBLE” y poder evaluar el comportamiento de cada caso según las transiciones efectuadas.

En la Figura 75 se observa que en el diagrama MQTTSN_ESTAB se realizó la transición de arribo de mensaje para pasar al estado “RECEPCION”, esto representa el arribo de un mensaje de establecimiento de conexión entre cliente y Gateway. Una vez que el software detecta que el diagrama está en el segundo estado, indica pintado de rojo la siguiente transición válida.

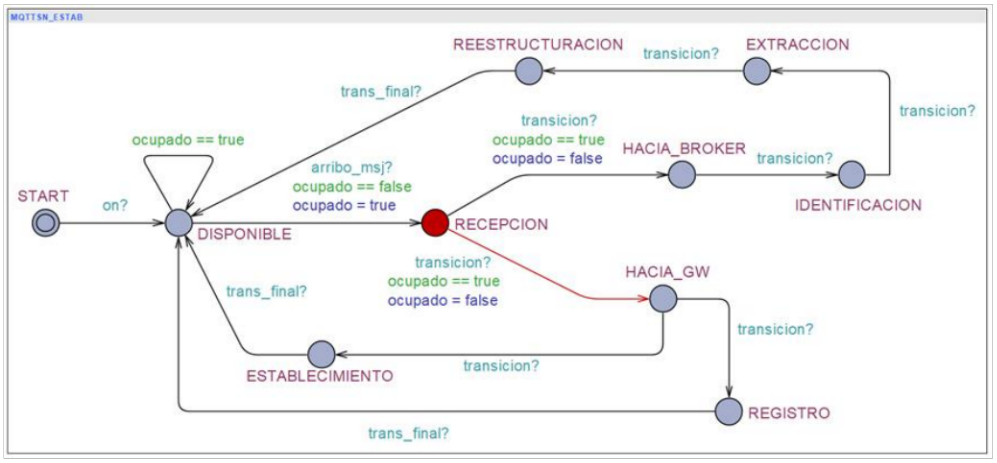


Figura 75: Transición hacia estado “HACIA_GW” y actualización de variable ocupado a “false” en el diagrama MQTTSN_ESTAB.

En la Figura 76 y Figura 77 se puede constatar que cuando un mensaje se encuentra en procesamiento en el diagrama MQTTSN_ESTAB, el software detecta el cambio de la variable ocupado a “true” por lo que no permite una transición fuera del estado “DISPONIBLE” en los diagramas MQTTSN_REGIST y MQTTSN_A_BROKERMQTT. Con esta condición se valida que el diagrama de estados reciba un mensaje y realice el procesamiento especifico antes de recibir otro.

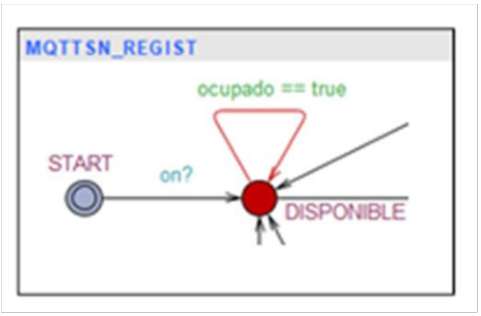


Figura 76: Transición de permanencia en estado “DISPONIBLE” mientras la variable ocupado tenga el valor “true” en el diagrama MQTTSN_REGIST.

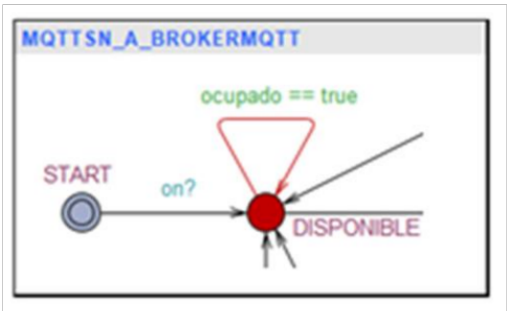


Figura 77: Transición de permanencia en estado “DISPONIBLE” mientras la variable ocupado tenga el valor “true” en el diagrama MQTTSN_A_BROKERMQTT.

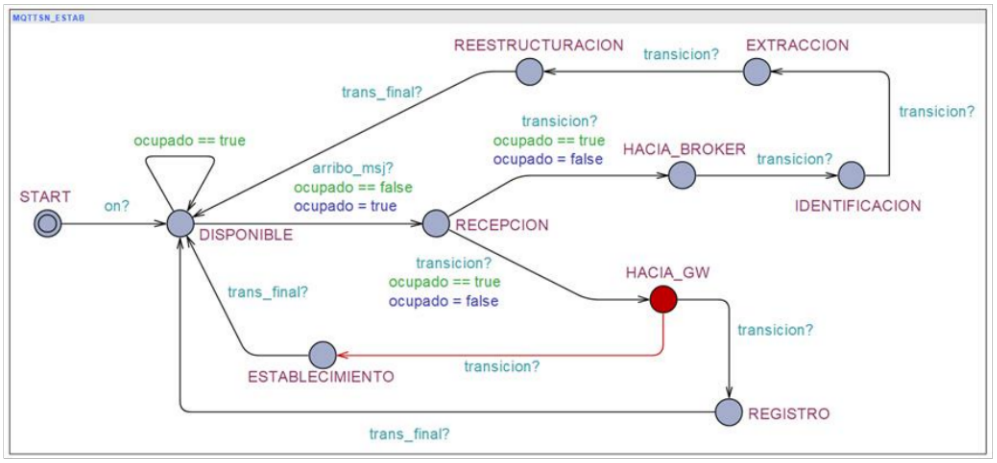


Figura 78: Transición hacia estado “ESTABLECIMIENTO” en el diagrama MQTTSN_ESTAB.

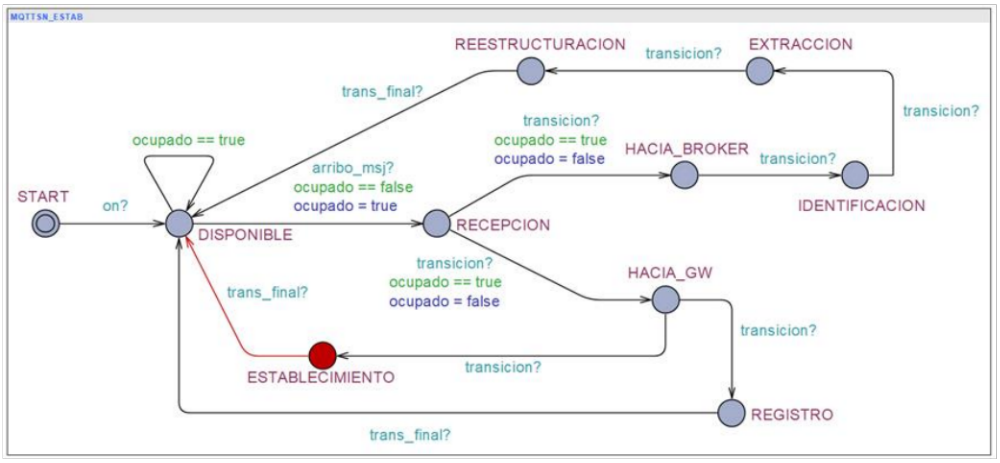


Figura 79: Transición final hacia estado “DISPONIBLE” en el diagrama MQTTSN_ESTAB.

En la Figura 80 se observa que en el diagrama MQTTSN_REGIST se realizó la transición de arribo de mensaje para pasar al estado “RECEPCION” debido a que la variable de ocupado paso a “false” por lo que el mensaje de registro pudo salir del estado “DISPONIBLE”. Así mismo el software detecta que es un mensaje enviado desde el cliente hacia el Gateway por lo que resalta en rojo la siguiente transición válida.

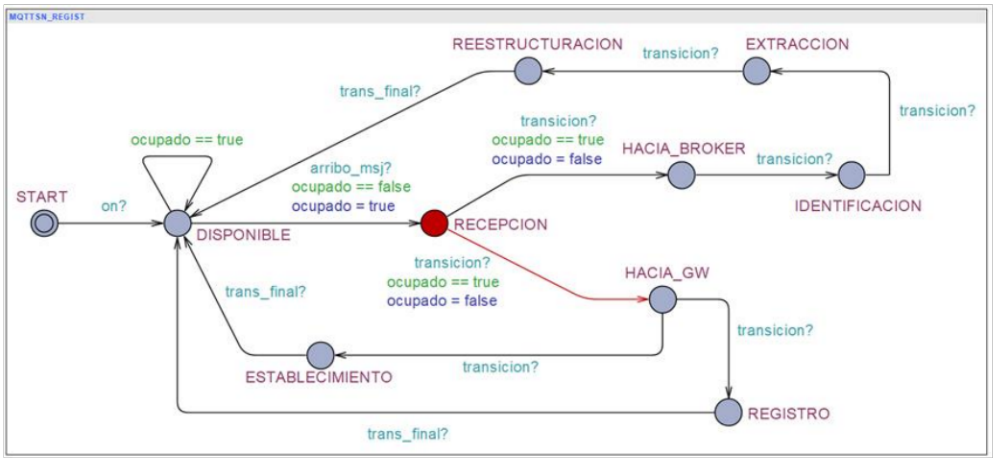


Figura 80: Transición hacia estado “HACIA_GW” y actualización de variable ocupado a “false” en el diagrama MQTTSN_REGIST.

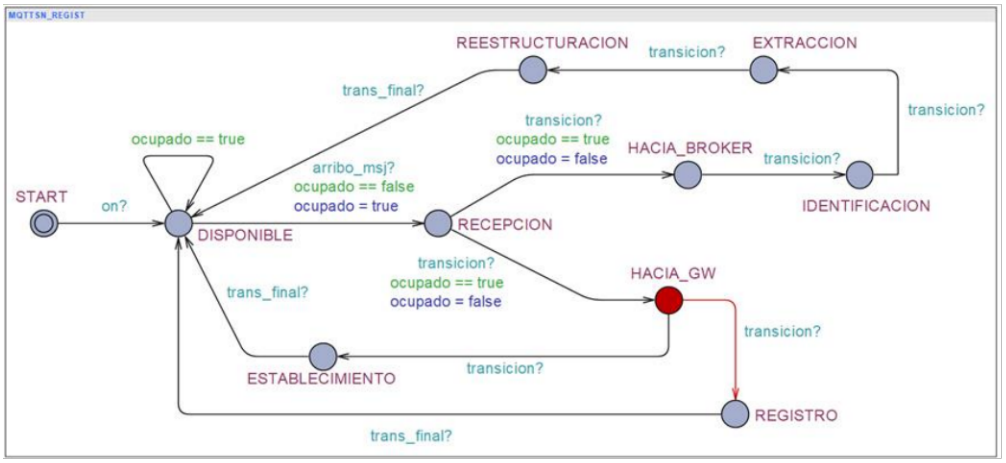


Figura 81: Transición hacia estado “REGISTRO” en el diagrama MQTTSN_REGIST.

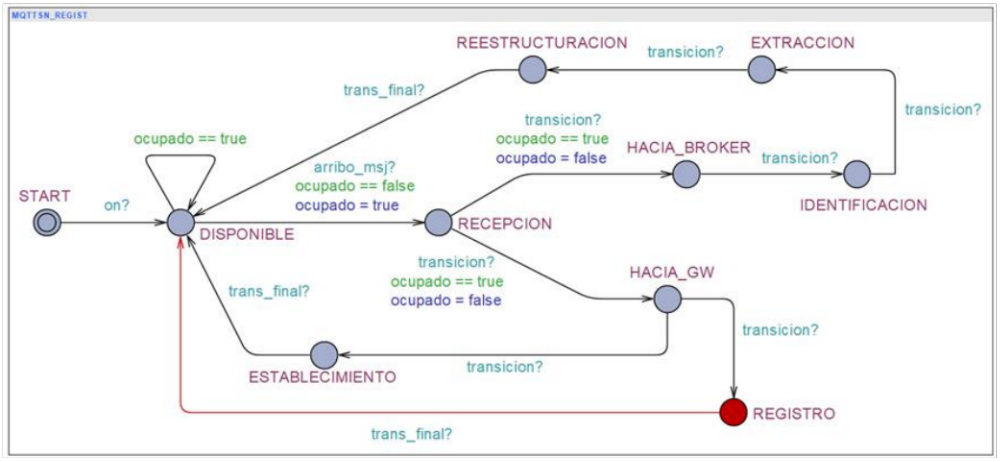


Figura 82: Transición final hacia estado “DISPONIBLE” en el diagrama MQTTSN_REGIST.

En la Figura 83 se observa que en el diagrama MQTTSN_A_BROKERMQTT después de que variable de ocupado cambio su valor, se realizó la transición de arribo de mensaje para pasar al estado “RECEPCION”. El software detecta que en este caso se simula la llegada de un mensaje del protocolo MQTT-SN dirigido hacia el bróker por lo que resalta en rojo la siguiente transición valida.

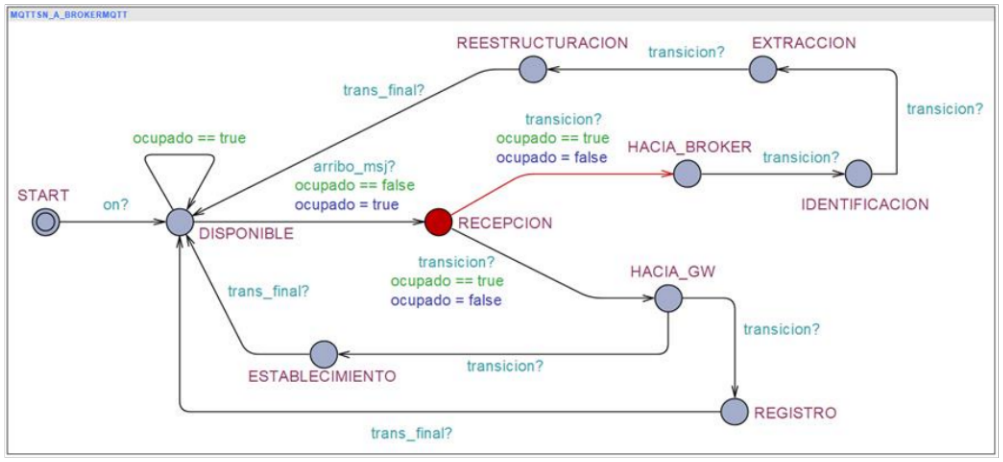


Figura 83: Transición hacia estado “HACIA_BROKER” y actualización de variable ocupado a “false” en el diagrama MQTTSN_A_BROKERMQTT.

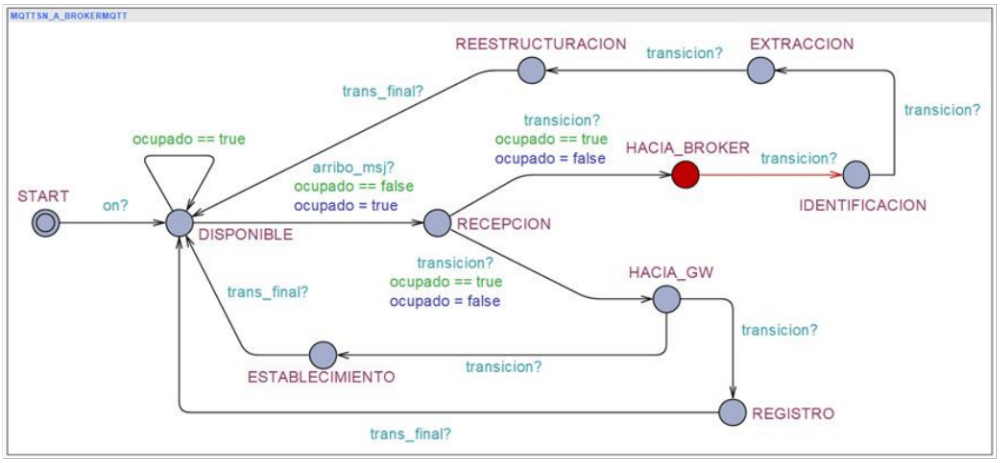


Figura 84: Transición hacia estado “IDENTIFICACION” en el diagrama MQTTSN_A_BROKERMQTT.

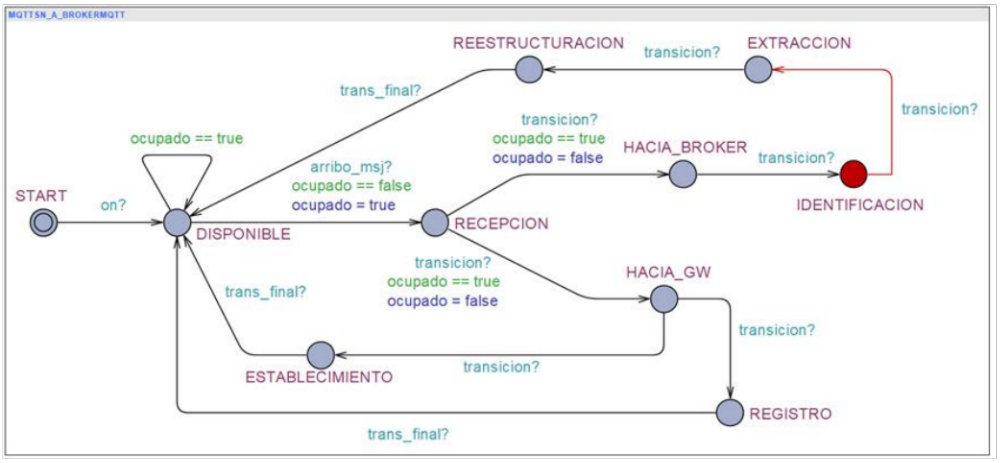


Figura 85: Transición hacia estado “EXTRACCION” en el diagrama MQTTSN_A_BROKERMQTT.

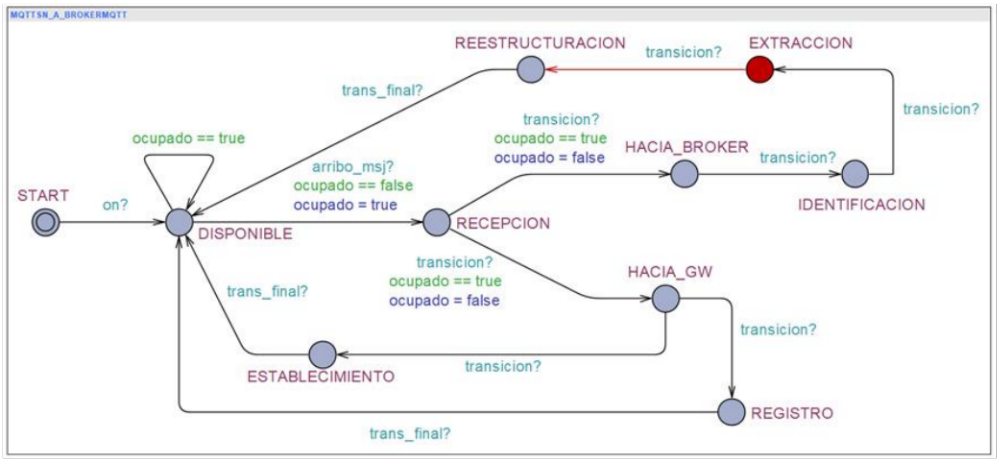


Figura 86: Transición hacia estado “REESTRUCTURACION” en el diagrama MQTTSN_A_BROKERMQTT.

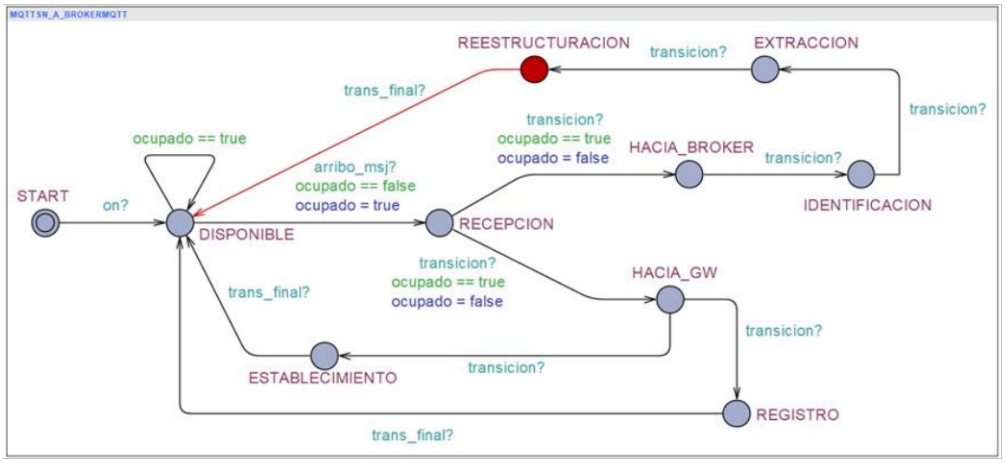


Figura 87: Transición final hacia estado “DISPONIBLE” en el diagrama MQTTSN_A_BROKERMQTT.

3.2 Validación del comportamiento del diagrama de estados del convertidor de protocolo MQTT a MQTT-SN

En esta sección se realiza la validación del diagrama de estados del convertidor de protocolo MQTT a MQTT-SN en UPPAAL. Debido a que en la comunicación desde el bróker hacia el nodo cliente MQTT-SN no requiere de una conexión previa con el Gateway, el sistema implementado en el software solo contiene un diagrama de estados el cual tiene el nombre MQTT_MQTTSN.

Se realiza la primera transición para iniciar en el estado “DISPONIBLE” y así poder empezar el análisis de los siguientes cambios de estado.

En la Figura 88 se observa que en el diagrama MQTT_MQTTSN se realizó la transición de arribo de mensaje para pasar al estado “IDENTIFICACION”. El software detecta que en este caso se simula la llegada de un mensaje del protocolo MQTT dirigido hacia el nodo MQTT-SN por lo que resalta en rojo la siguiente transición valida

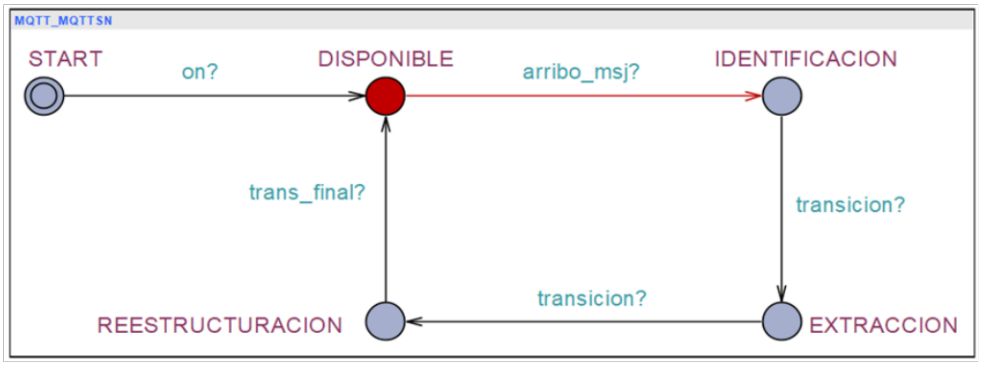


Figura 88: Transición hacia estado “IDENTIFICACION” en el diagrama MQTT_MQTTSN.

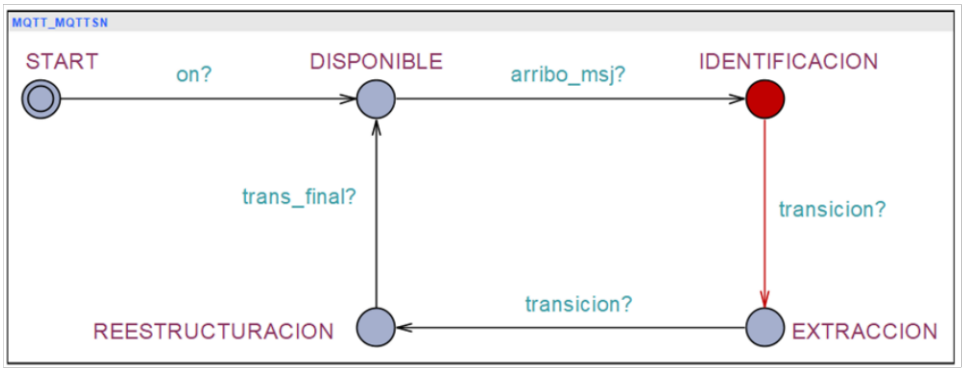


Figura 89: Transición hacia estado “EXTRACCION” en el diagrama MQTT_MQTTSN.

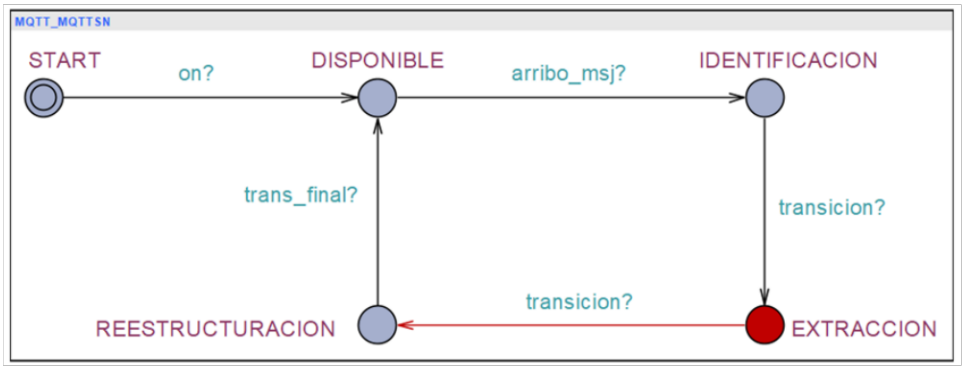


Figura 90: Transición hacia estado “REESTRUCTURACION” en el diagrama MQTT_MQTTSN.

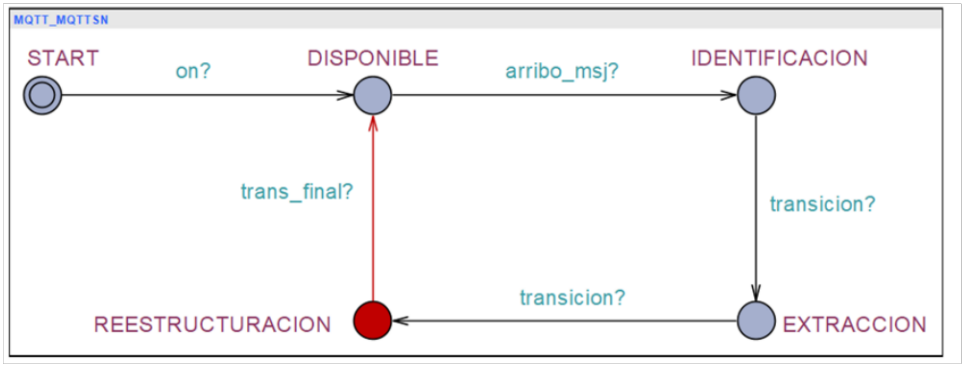


Figura 91: Transición hacia estado “DISPONIBLE” en el diagrama MQTT_MQTTSN

4 CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

En el presente trabajo de integración curricular se realizó la implementación del diagrama de estados del convertidor de protocolos MQTT-SN a MQTT utilizando UML cumpliendo con todos los objetivos planteados.

Después de haber realizado este trabajo, se puede concluir lo siguiente:

- Se desarrolló una base teórica y operativa de los protocolos MQTT y MQTT-SN enfocada a ser una referencia para la futura implementación del convertidor de protocolos en un nodo Gateway para una red inalámbrica de sensores con un servidor MQTT.
- Se realizó el análisis de los campos presentes en los mensajes definidos en cada protocolo para poder establecer la correspondencia de datos en el proceso de conversión. Además, se generó una guía gráfica de la correspondencia de campos en cada caso de conversión que lo ameritaba.
- La conversión del protocolo MQTT-SN a MQTT requiere procesos previos para establecer un canal de comunicación entre el nodo sensor y el nodo Gateway debido a las limitaciones en la capacidad de transmisión del protocolo MQTT-SN, por lo que hay mensajes específicos definidos en este protocolo que no tienen un equivalente en el protocolo MQTT.
- Los diagramas de estados diseñados en el presente trabajo representan el funcionamiento del convertidor de protocolos en ambos sentidos, especificando los procesos realizados en cada estado, los eventos que originan las diferentes transiciones y los resultados obtenidos a la salida.
- Los diagramas de estados están desarrollados en UML para ser una guía en una futura implementación del convertidor de protocolos en código utilizando cualquier lenguaje de programación orientado a objetos.
- La prueba de verificación del diagrama de estados del convertidor de protocolo MQTT-SN a MQTT comprueba el correcto funcionamiento de este, simulando el arribo de diferentes tipos de mensajes por lo que al no presentar inconsistencias puede ser implementado en una aplicación.
- La prueba de verificación del diagrama de estados del convertidor de protocolo MQTT a MQTT-SN no requirió la definición de condiciones en ninguna transición debido a que todos los mensajes enviados por el bróker hacia el nodo sensor siguen el mismo flujo de estados.

4.2 Recomendaciones

- Es recomendable seguir la guía del esquema de correspondencia de campos en la conversión de protocolos para entender de mejor manera el tratamiento de la información contenida en cada campo para los diferentes mensajes que ingresan al proceso de conversión.
- Es recomendable para un futuro trabajo de implementación del convertidor de protocolo MQTT-SN a MQTT, tomar en cuenta para la programación que los pseudoestados de decisión definidos en este trabajo se utilizan para ejemplificar claramente que en ese sentido de conversión se requieren procesos previos con mensajes específicos asignados. Por esta razón, se debe realizar un tratamiento a los datos de ingreso para poder destinar cada mensaje en su debido procesamiento.
- El software UPPAAL puede ser utilizado para validar el comportamiento de los diagramas de estado individualmente, pero se recomienda simular un sistema compuesto por tres diagramas para verificar una interoperabilidad de todas las transiciones definidas en el diagrama de comportamiento del mensaje, pero en tres escenarios distintos.

5 REFERENCIAS BIBLIOGRÁFICAS

[1]. P. Desai, A. Sheth and P. Anantharam. (2015). “Semantic Gateway as a Service Architecture for IoT Interoperability,” in IEEE International Conf. on Mobile Services. [En línea]. Disponible: https://ieeexplore.ieee.org/document/7226706

[2]. A. Stanford-Clark and H. Linh. (2013, Nov 14). “MQTT For Sensor Networks (MQTT-SN) Protocol Specification” (Ver. 1.2). [En línea]. Disponible: https://www.oasis-open.org/committees/download.php/66091/MQTT-SN_spec_v1.2.pdf

[3]. International Business Machines Corporation (IBM) and Eurotech. “MQTT Protocol Specification” (Ver. 3.1). [En línea]. Disponible: https://public.dhe.ibm.com/software/dw/webservices/ws-mqtt/mqtt-v3r1.html

[4]. N. Medvidovic, D. Rosenblum, J. Robbins and D. Redmiles. (2000, Aug 20). “Modeling Software Architectures in the Unified Modeling Language”. [En línea]. Disponible: https://www.ics.uci.edu/~redmiles/publications/J007-MRR+02.pdf

[5]. N. Aakvaag and J. Frey. (2006). “Redes de sensores inalámbricos”. [En línea]. Disponible: http://www.ie.com.co/pdf/ABB/02-2006/39-42%25202M631_SPA72dpi.pdf

[6]. N. Ahmed, H. Rahman and I. Hussain. (2016). “A comparison of 802.11ah and 802.15.4 for IoT”. [En línea]. Disponible: https://www.sciencedirect.com/science/article/pii/S2405959516300650

[7]. J. Wu and I. Stojmenovic. (2004). “Ad Hoc Networks”. [En línea]. Disponible: https://www.cse.fau.edu/~jie/research/publications/Publication_files/adhoc.pdf

[8]. IEEE, (2006). “IEEE. 802.15.4a-2006, Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)”. [En línea]. Disponible: https://ieeexplore.ieee.org/document/1700009?arnumber=1700009

[9]. S. Sinche. (2020, Jan 27). “New Models of Reliability in the New Generation of Internet of Things”. [En línea]. Disponible: https://eg.uc.pt/handle/10316/91095

[10]. S. Sinche, D. Raposo, N. Armando, A. Rodrigues, F. Boavida, V. Pereira and J. Sá Silva. “A Survey of IoT Managmenent Protocols and Frameworks”, in IEEE Communications Surveys & Tutorials, vol. 22, no. 2, pp. 1168-1190, Secondquarter 2020, doi: 10.1109/COMST.2019.2943087.

[11]. U. Hunkeler, H. L. Truong, and A. Stanford-Clark, “MQTT-S – A Publish/Subscribe Protocol For Wireless Sensor Networks,” in Communication systems software and middleware and workshops, 2008. comsware 2008. 3rd international conference on. IEEE, 2008. [En línea]. Disponible: https://sites.cs.ucsb.edu/~rich/class/cs293b-cloud/papers/mqtt-s.pdf

[12]. V. Lahoz. (2019, Feb 1). “El protocolo MQTT: impacto en España”. [En línea]. Disponible: https://www.securityartwork.es/2019/02/01/el-protocolo-mqtt-impacto-en-espana/

[13]. D. Soni and A. Makwana. (2017). “A survey on MQTT: A protocol of internet of things (IoT)”. [En línea]. Disponible: https://www.researchgate.net/profile/Dipa-Soni/publication/316018571_A_SURVEY_ON_MQTT_A_PROTOCOL_OF_INTERNET_OF_THINGSIOT/links/58edafd4aca2724f0a26e0bf/A-SURVEY-ON-MQTT-A-PROTOCOL-OF-INTERNET-OF-THINGSIOT.pdf

[14]. P. Egli. (2017). “MQTT – An introduction to MQTT, a protocol for m2m and IoT applications”. [En línea]. Disponible: http://peteregli.net/content/iot/MQTT/MQTT.html

[15]. MQTT Version 3.1.1, OASIS Standard, A. Banks and R. Gupta, 2014. [En línea]. Disponible: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.pdf

[16]. K. Jia, J. Xiao, S. Fan and G. He. (2018, Jun 25). “A MQTT/MQTT-SN-Based User Energy Management System for Automated Residential Demand Response: Formal Verification and Cyber-Physical Performance Evaluation”. [En línea]. Disponible: https://www.mdpi.com/2076-3417/8/7/1035

[17]. J. Rumbaugh, I. Jacobson and G. Booch, “Un paseo por UML,” in El Lenguaje Unificado de Modelado. Manual de Referencia. Madrid, 2000, pp. 21-34. [En línea]. Disponible: https://ingenieriasoftware2011.files.wordpress.com/2011/07/el-lenguaje-unificado-de-modelado-manual-de-referencia.pdf

[18]. M. C. Otero, “EVALUACIÓN EMPÍRICA DE LA COMPRENSIÓN DEL MODELADO DINÁMICO EN LOS LENGUAJES UML Y OML DE APLICACIONES SOFTWARE”, Ph.D. dissertation, Dept. Inf. Eng., Universidad del país vasco, Donostia, 2003. [En línea]. Disponible: http://www.vc.ehu.es/jiwotvim/Tesis_MCO.pdf

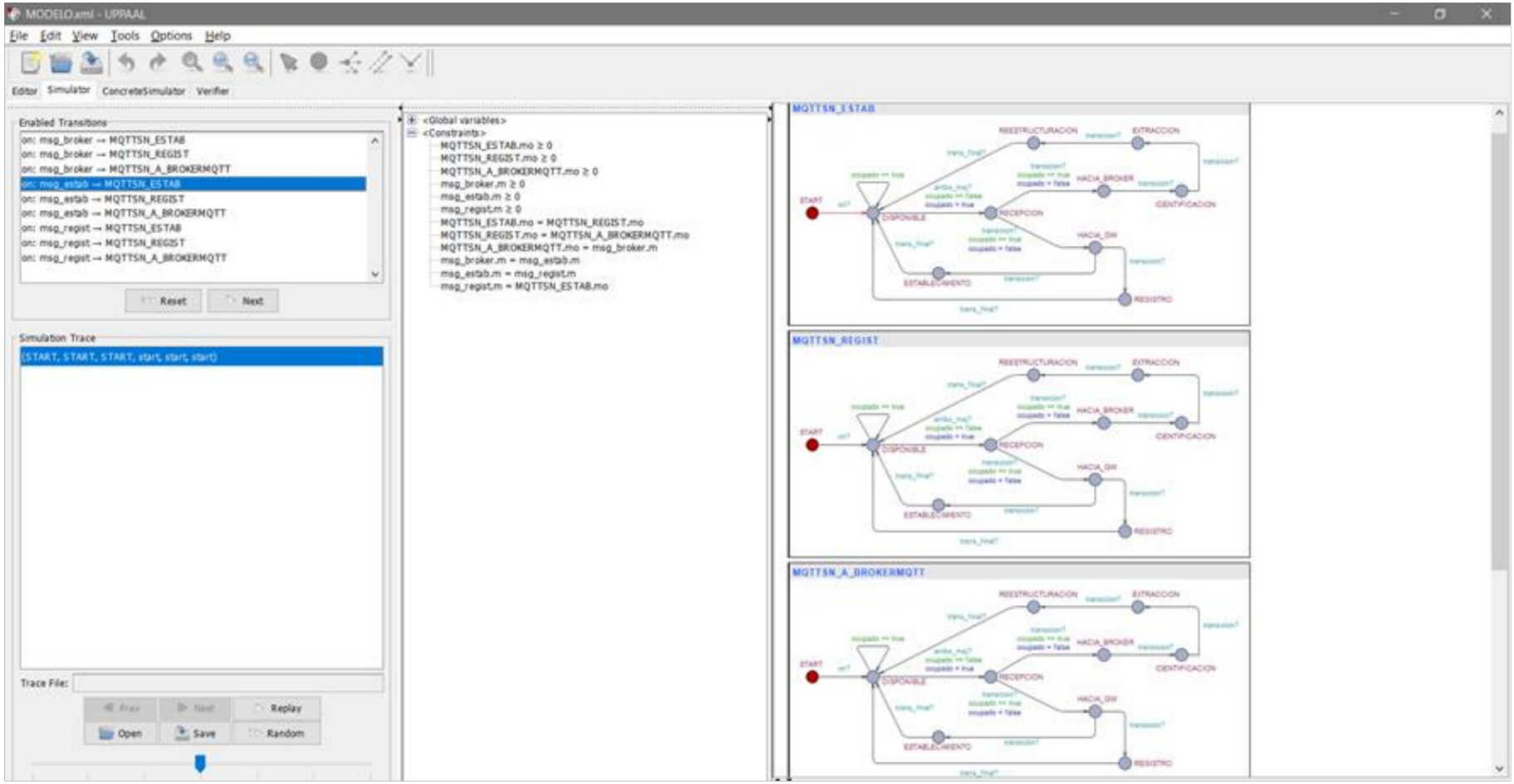
[19]. D. Mantilla and A. Santos. (2007). “Desarrollo de un portal web para el ingreso y consultas de notas para el colegio nacional mixto María Angélica Carrillo de Mata Martínez”. [En línea]. Disponible: https://bibdigital.epn.edu.ec/bitstream/15000/1359/1/CD-0793.pdf

[20]. UPPAAL. (2019, Nov 18). [En línea]. Disponible: https://uppaal.org/

6 ANEXOS

ANEXO I. Captura de pantalla de la ventana de simulación completa del software UPPAAL. Contiene los tres diagramas de estados simulados en la parte derecha y las transiciones disponibles para cada uno en la parte izquierda.

ANEXO I


MODELO.xml - UPPAAL
File Edit View Tools Options Help
Editor Simulator ConcreteSimulator Verifier
Enabled Transitions
on: msg_broker → MQTTSN_ESTAB
on: msg_broker → MQTTSN_REGIST
on: msg_broker → MQTTSN_A_BROKERMQTT
on: msg_estab → MQTTSN_ESTAB
on: msg_estab → MQTTSN_REGIST
on: msg_estab → MQTTSN_A_BROKERMQTT
on: msg_regist → MQTTSN_ESTAB
on: msg_regist → MQTTSN_REGIST
on: msg_regist → MQTTSN_A_BROKERMQTT
Reset
Next
Simulation Trace
(START, START, START, start, start, start)
Trace File:
Prev
Next
Replay
Open
Save
Random
<Global variables>
<Constraints>
MQTTSN_ESTAB.mo ≥ 0
MQTTSN_REGIST.mo ≥ 0
MQTTSN_A_BROKERMQTT.mo ≥ 0
msg_broker.m ≥ 0
msg_estab.m ≥ 0
msg_regist.m ≥ 0
MQTTSN_ESTAB.mo = MQTTSN_REGIST.mo
MQTTSN_REGIST.mo = MQTTSN_A_BROKERMQTT.mo
MQTTSN_A_BROKERMQTT.mo = msg_broker.m
msg_broker.m = msg_estab.m
msg_estab.m = msg_regist.m
msg_regist.m = MQTTSN_ESTAB.mo
MQTTSN_ESTAB
MQTTSN_REGIST
MQTTSN_A_BROKERMQTT
REESTRUCTURACION
EXTRACCION
HACIA_BROKER
IDENTIFICACION
RECEPCION
HACIA_GW
ESTABLECIMIENTO
REGISTRO
DISPONIBLE
START