



Universidad autónoma de baja california

Ingeniería en computación

Internet de las cosas

Taller resumen IoT

Aguilar Noriega Leocundo

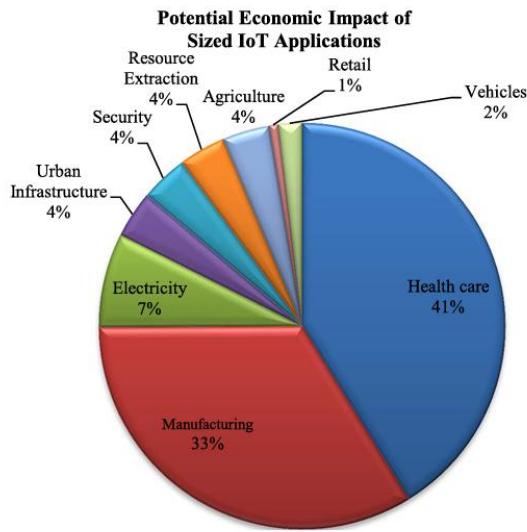
Garcia Chaves erik

10 de febrero del 2026

- I. **Introducción:** los dispositivos IoT cada vez se están introduciendo más a la vida de las personas con aspectos que hacen la vida un poco más fácil, como las llamadas “casas inteligentes”, el IoT transforma objetos tradicionales en inteligentes aprovechando sus tecnologías subyacentes. Como la computación **ubicua y omnipresente, tecnologías de la comunicación, redes de sensores, protocolos de internet y aplicaciones.** la computación ubicua y los servicios analistas forman servicios independientes del dominio de aplicación. La estandarización de la arquitectura puede considerarse la columna vertebral de IoT para crear un entorno competitivo que permita a las empresas ofrecer productos de calidad. La arquitectura actual de internet puede ser un desafío en el año 2010 el numero de objetos conectados a internet supera la población humana, por lo que el empezar a utilizar un espacio de direcciones más amplio se vuelve necesario (como el IPv6) se vuelve necesario para satisfacer las necesidades de los clientes. El otro aspecto importante es la privacidad y la seguridad debido a la **heterogeneidad** inherente de los objetos conectados a internet y la capacidad de monitorear y controlar objetos físicos.
- El costo de estos dispositivos es un punto igual de importante estos productos deben ser rentables tanto para que el fabricante como los clientes, para que sea rentable su utilización.
- Este artículo está dividido en diferentes bloques los cuales estarán tratando diferentes temas, tenemos a la sección 2 que va a enfocarse en oportunidad de mercado, la 3 y 4 discuten la arquitectura general del IoT. La sección 5 los protocolos y estándares actuales de la IoT, en la sección 6 la seguridad y confianza, la monitorización, la gestión y aspectos de calidad de servicio. La sección 7 la interacción entre el big data y el IoT y la necesidad de gestionar y analizar grandes cantidades de datos generados por el IoT. La sección 8 la necesidad de servicios inteligentes del IoT para lograr una mejor integración horizontal entre los servicios de IoT. La 9 la integración de diferentes protocolos del IoT para ofrecer funcionalidades deseadas mediante algunos casos de uso. La sección 10 presenta un resumen

- II. Oportunidad de mercado:
- El IoT ofrece una gran oportunidad de mercado para los fabricantes, proveedores de servicios de internet. Se espera un estimado de 212 mil millones de entidades desplegadas a nivel mundial. Para el 2022 se espera que el tráfico de M2M (machine-to-machine) conste hasta el 45% de todo el tráfico de internet.
- Los servicios de IoT tendrán un impacto considerable en aplicaciones de salud y manufactura.

Se espera un crecimiento anual de 1.1 – 2.5 mil millones de dólares en la economía mundial en aplicaciones de atención media y servicios relacionados con el IoT. En la siguiente imagen se muestra la participación de mercado proyectada de las aplicaciones dominantes de IoT



Estas estadísticas apuntan a un crecimiento potencialmente significativo y rápido del IoT en el futuro cercano, así como de las industrias y servicios relacionados. Este progreso ofrece una oportunidad para que los fabricantes de equipos tradicionales transformen sus productos en “cosas inteligentes”. Ofreciendo un servicio pensando en la combinación de flujos de persona-maquina (P2M) maquina-maquina (M2M) y de persona-persona (P2P).

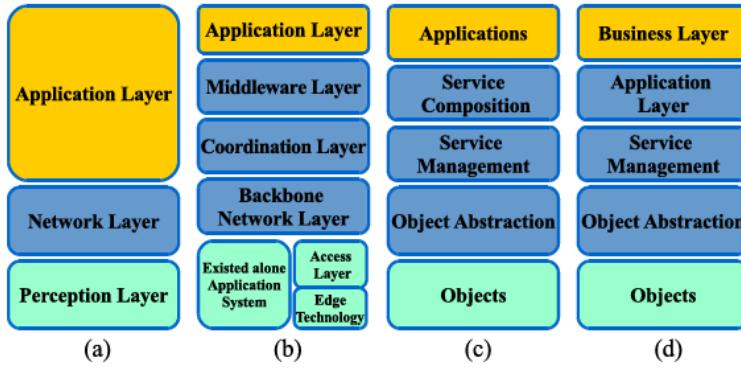
III. Arquitectura del IoT:

El IoT debería de ser capaz de conectar as miles de millones de dispositivos por lo que hay una necesidad critica de una arquitectura flexible y en capas.

Aun no se ha convertido a un modelo de referencia. Algunos proyectos como **IoT-A** intentan diseñar una arquitectura común basada en el análisis de las necesidades de los investigadores y la industria.

Del conjunto propuesto el modelo básico es una arquitectura de 3 capas, que consta de las capas de **aplicación, red y percepción**.

En la siguiente imagen se muestra algunas arquitectura comunes de un intento de mejorar el diseño base.



A. Capa de objetos < **object layer** >:

es la capa de dispositivos o de percepción. Representa los sensores, tiene como objetivo recopilar y procesar información. En otras palabras, es la capa que está en contacto directo con el mundo real y recopila toda esa información para después en un momento transformarlo en datos para el dispositivo. Se deben utilizar mecanismos estandarizados de tipo **plug-and-play** para configurar objetos heterogéneos.

B. Capa de abstracción de objetos < **object abstraction layer** >

Transfiere datos producidos por la capa de objetos a la capa de gestión de servicios a través de canales seguros. Estos se pueden transferir mediante diferentes tecnologías como pueden ser **RFID, 3G, GSM, UMTS, WIFI, Bluetooth de baja energía, infrarrojo, ZigBee**.

C. capa de gestión de servicios < **service management layer** >

esta capa empareja un servicio con su solicitante en función de direcciones y nombres, es la capa que procesa la información objetiva del mundo real que se transformó en datos, es procesada y en base a eso toma decisiones que después eran entregado por protocolos de red.

D. Capa de aplicación < **application layer** >

Proporciona los servicios solicitados por los clientes. Es la capa visible, con lo que el usuario estará interactuando con sus dispositivos, lo que abstracto el sistema del mundo real lo plasma en una aplicación que el usuario puede ser y tomar decisiones en el mundo real.

E. Capa de negocios < **business layer** >

Administra las actividades y servicios generales del sistema. Su propósito es construir un modelo de negocio, gráficos, diagramas. Etc. de los datos recibidos de la capa de aplicación. Esta capa permite apoyar los procesos de toma de decisiones basado en el análisis de big data. Esta capa

supervisa las otras 4. Esta capa compara la salida de cada capa con la esperada para mejorar los servicios y mantener la privacidad de los usuarios.

En el modelo de 5 capas la de aplicación es la interfaz mediante la cual los usuarios pueden interactuar con el dispositivo y consultar datos. Proporciona una interfaz con la capa de negocios donde se pueden producir análisis e informes de alto nivel.

IV. Elementos IoT

Los elementos del IoT nos ayudan a comprender mucho mejor el cómo estos aparatos funcionan, tenemos un diagrama que nos puede dar una idea visual.



A. Identificación:

La identificación es crucial para el IoT para nombrar y emparejar servicios con su demanda. El **ID** hace referencia al nombre del objeto y la **dirección** hace referencia a su dirección de una red de comunicaciones. Los métodos de direccionamiento incluyen **IPv6** e **IPv4**. **6LoWPAN** hace que el direccionamiento de **IPv6** sea adecuado para redes inalámbricas de baja potencia.

El ID ayuda a identificar el dispositivo dentro de una red, dentro de una cada dispositivo conectado a la red tendrá un **ID** único mientras que la dirección de la red podría tener una IP publica o no privada.

B. Sensación:

En el IoT, los dispositivos recopilan datos que son enviado a una base de datos, a la nube, por ejemplo, para recolectar los datos y analizarlos, con estos datos el dispositivo puede tomar acciones dependiendo de lo que perciba o el usuario lo podría hacer. Para poder tomar estos datos, se utilizan placas denominadas **SBC** computadora de placa única, la más común puede ser la **rasberry pi**, la cual es una microcomputadora con funcionalidades **TCP/IP integradas** para poder realizar productos de IoT con la capacidad de comunicación en internet.

C. Comunicación:

Los dispositivos IoT comúnmente operan en presencia de señales con presencia de ruido o de baja potencia, como lo puede ser **WIFI, Bluetooth, IEEE 802.15.4, Z-wave, el RFID o el NFC**. Las etiquetas **RFID** funcionan para dar una identificación a un objeto, esos dispositivos pueden ser pasivos, una etiqueta con una antena que no contiene batería ni nada. Activas, o semipasivas/activas. El protocolo NFC funciona en una banda de alta frecuencia a 13.56 MHz y admite una velocidad de transmisión de datos de 424 kbps. Banda ultra ancha (**UWB**), funciona para soportar comunicación en un área de corto alcance utilizando baja energía y alto ancho de banda. El wifi funciona para intercambiar datos entre dispositivos en un rango de 100 m, el cual suele ser el más común. El estándar IEEE 802.15.4 especifica tanto la capa física como un control de acceso al medio para redes inalámbricas de baja potencia.

D. Computación.

Las unidades de procesamiento y las aplicaciones de software representan el cerebro y la capacidad de cálculo del IoT. Un dispositivo IoT necesita un sistema operativo de tiempo real (RTOS) el cual es lo que hará funcionar de manera correcta todo el software para estos existen varias plataformas que nos ofrecen esta solución. Tenemos **RTOS cantiki, TinyOS, LiteOS, Riot OS, Google junto con Android planean llevar este SO al IoT de los vehículos**. Otro miembro importante del ecosistema de una arquitectura IoT es la nube, en donde estos dispositivos pueden subir la información que capturaron para que los grandes volúmenes de datos se puedan procesar en tiempo real.

E. Servicios:

Los servicios de IoT se pueden categorizar en 4 clases **servicio de identidad-relación, servicio de agregación de información, servicios conscientes de la colaboración y servicios ubicuos**.

El servicio de identidad-relación es el más básico e importante servicio usado en otros tipos de servicios. Toda aplicación necesita relacionar los objetos reales con los objetos sus objetos del mundo virtual.

El servicio de recolección de información, recopila y resumen mediciones sensoriales sin procesar que deben ser procesadas y reportarlas a la aplicación IoT.

Los servicios conscientes de la colaboración actúan sobre la información que el servicio de recolección de información obtiene, esto para tomar decisiones y reaccionar en consecuencia.

Los servicios ubicuos tienen como objetivo el siempre poder proporcionar servicios de colaboración en cualquier momento a quien lo necesite.

El objetivo de todas las aplicaciones IoT es el servicio ubicuo, pero llegar a este nivel resulta difícil en estos tiempos.

Los servicios de IoT casas, edificios, la industria, transportes, salud, redes, ciudades, etc. En casi toda industria o aspecto de la vida el IoT puede incorporarse esto con el propósito de hacer la vida más fácil, el evitar accidentes, el que cada sistema sea mucho más seguro como el transporte y la salud que es donde un error puede costar vidas humanas, los sistemas autónomos y redes de IoT pueden reducir estos percances.

F. Semántica:

La semántica en IoT hace referencia a la capacidad de extraer conocimiento de manera inteligente por diferentes máquinas para proporcionar los servicios requeridos. Esto incluye el descubrir y utilizar recursos y modelar información, reconocer, analizar datos para dar sentido a una decisión. Este requerimiento es apoyado por tecnologías de la web semántica como el marco de descripción de recursos (RDF) y el lenguaje de ontologías web (OWL).

EXI es importante en el contexto del IoT porque está diseñado para optimizar las aplicaciones XML en entornos con recursos limitados. Además, reduce las necesidades de ancho de banda sin afectar recurso relacionados como la batería, el tamaño del código, la energía consumida y el tamaño de la memoria.

V. Estándares de comunicación IoT

En la siguiente tabla se muestra un resumen de los protocolos más destacados definidos por los diferentes grupos encargados de regular los estándares de comunicación.

Application Protocol	DDS	CoAP	AMQP	MQTT	MQTT-SN	XMPP	HTTP REST
Service Discovery	mDNS		DNS-SD				
Infrastructure Protocols	Routing Protocol	RPL					
	Network Layer	6LoWPAN			IPv4/IPv6		
	Link Layer	IEEE 802.15.4					
	Physical/Device Layer	LTE-A	EPCglobal	IEEE 802.15.4	Z-Wave		
Influential Protocols	IEEE 1888.3, IPSec				IEEE 1905.1		

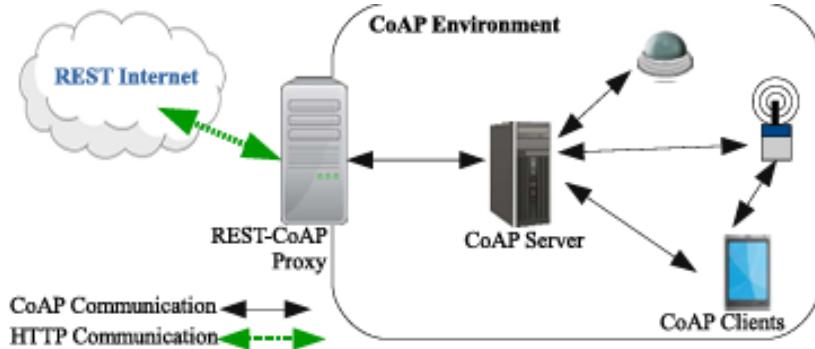
Se clasifican los protocolos IoT en **4 categorías, protocolos de aplicación, protocolos de descubrimiento de servicios, protocolos de infraestructura y otros protocolos influyentes**.

No todos estos protocolos deben ser empleados para desarrollar la aplicación IoT, dependiendo la aplicación de esta se usarán unos u otros.

A. Protocolo de aplicación.

- 1) Protocolo de aplicación restringida (**CoAP**): define un protocolo de transferencia web basado en **representational state transfer (REST)** sobre las funcionalidades **HTTP**. **REST** representa una funcionalidad más fácil de intercambiar datos entre cliente servidor a través de **HTTP**. Este protocolo se basa en una arquitectura cliente-servidor sin estado. Se utiliza en aplicaciones móviles y de redes sociales, mediante el uso de los métodos **HTTPS get, post, put, delete**. **REST** permite consumir servicios web como el **protocolo simple de acceso a objetos (SOAP)**. Pero utilizando **identificadores uniformes de recursos (URLs)** como sustantivo y los métodos **HTTP** como verbos. A diferencia de **REST**, **CoAP** está ligado a **UDP** por defecto lo que lo hace más adecuado para IoT. Además, **CoAP** modifica algunas funcionalidades de **HTTP** para cumplir con los requisitos del IoT como el bajo consumo de energía y la operación en presencia de enlaces ruidosos y propensos a perdidas.

CoAP está basado en **REST** por lo que la conversión entre estos es sencilla. Se muestra una ilustración de las funcionalidades de este protocolo.



CoAP tiene como objetivo que dispositivos de bajos recursos computacionales utilicen interacciones **RESTful**. **CoAP** puede dividirse en 2 capas. **Subcapa de mensajería y la subcapa de solicitud/respuesta**. La **subcapa de mensajería**, detecta duplicaciones y proporciona una comunicación confiable sobre la capa de transporte **UDP**. **UDP** no tiene un mecanismo de recuperación de errores incorporado. La **subcapa de solicitud/respuesta** se encarga de las comunicaciones **REST**. **CoAP** utiliza 4 tipos de mensajes: **conformables, no conformables, de reinicio y de acuse de recibido (ACK)**. La fiabilidad de **CoAP** se logra mediante una combinación de mensajes confirmarlos y no confirmarlos. Se emplean 4 modos de respuesta como se ilustran:

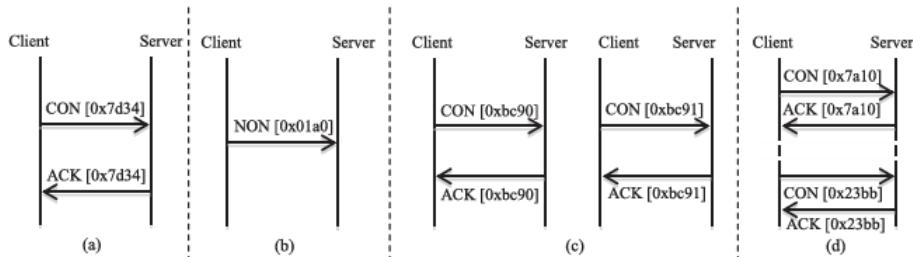


Fig. 6. CoAP message types [64]. (a) Confirmable. (b) Non-confirmable. (c) Piggybacked responses. (d) Separate response.

El **modo de respuesta separada** se utiliza cuando el servidor necesita esperar un tiempo específico antes de responder al cliente. **Modo de respuesta no confirmarle** el cliente envía datos sin esperar un **ACK** mientras que se utilizan IDs de mensaje para detectar duplicados. El lado del servidor contesta con un mensaje **RST** cuando se pierden mensajes u ocurren problemas de comunicación. **CoAP** utiliza métodos como **GET, PUT, POST, DELETE** para realizar operación de **crear, recuperar, actualidad y eliminar (CRUD)**.

CoAP utiliza un formato simple y pequeño para codificar mensajes. La primera parte fija es un encabezado de 4 bytes. Luego puede aparecer un valor token cuya longitud puede variar de 0 a 8 bytes. Este valor se usa para correlacionar solicitudes y respuestas. Las opciones y la carga útil son los siguientes campos opcionales. Un mensaje típico de **CoAP** puede contener de 10 a 20 bytes.

0	1	2	3	4	5	6	7	8	16	31	...
Ver	T	OC		Code				Message ID			
Token (if any)											
Options (if any)											
Payload (if any)											

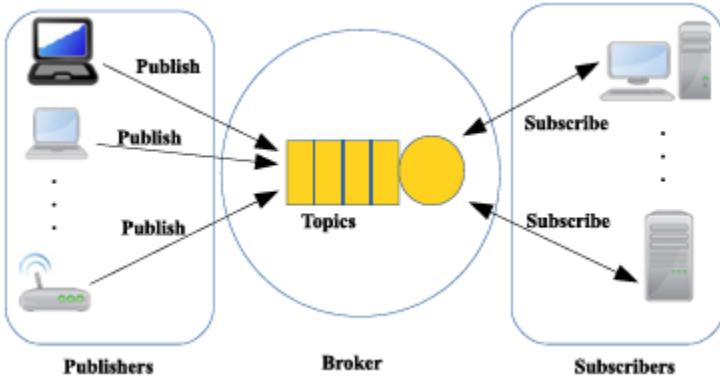
Los campos en el encabezado son, **ver** es la versión de **CoAP**, **T** es el tipo de transacción, **OC** el conteo de opciones, **code** representa el método de solicitud (1-10) o el código respuesta (40-255), **messege ID** de transacción en el encabezado es un identificador único para emparejar la respuesta.

Algunas de las funciones importantes proporcionadas por **CoAP** son:

- *Observación de recursos*: suscripciones bajo demanda para monitorear recursos de interés utilizando un mecanismo de publicación/suscripción
- *Transporte de recursos por bloques*: la habilidad de intercambiar datos entre el cliente y el servidor sin la necesidad de volver a cargar toda la información
- *Descubrimiento de recursos*: el servidor utiliza rutas **URI** bien conocidas basados en los campos de enlace web en el formato de enlace **CoRE** para proporcionar los recursos al cliente
- *Interacción con HTTP*: flexibilidad para comunicarse con varios dispositivos, **REST** permite que **CoAP** interactuar fácilmente con HTTP a través de un proxy.
- *Seguridad*: es un protocolo seguro basado en **DTLS** para garantizar la integridad y confidencialidad de los mensajes intercambiados.

2) Messegue Queue telemetry transport (**MQTT**):

MQTT tiene como objetivo conectar dispositivos y redes integradas con aplicaciones y middlewere. Utiliza un mecanismo de enrutamiento (**uno a uni**, **uno a muchos**, **muchos a muchos**), permite que **MQTT** sea un protocolo de conexión óptimo para el **IoT** y **M2M**. **MQTT** utiliza el patrón de publicación/suscripción para proporcionar flexibilidad en la transición y simplicidad de implementación.



Como se muestra en la imagen **MQTT** es ideal para dispositivos con recursos limitados que utilizan enlaces poco fiables o de bajo ancho de banda. Se basa en el protocolo **TCP** entrega mensajes a través de 3 niveles de **QoS**. Existen 2 especificaciones principales para **MQTT**: **MQTT v3.1** y **MQTT-SN**, esta última fue definida específicamente para redes de sensores y define un mapeo **UDP de MQTT** y añade soporte de bróker para indexación de nombres de temas. Las especificaciones proporcionan 3 elementos: **semántica de conexión, enrutamiento y puntos finales**.

MQTT consta de 3 componentes: **suscriptor, publicador y bróker** (**bróker: el manejador, los dispositivos nunca hablan entre si el bróker es el que se encarga de recibir y enviar a quien es el destinatario de ese mensaje.**) un dispositivo interesado en temas específicos se registra como suscriptor para ser informador por el bróker cuando se hagan esas publicaciones. **El publicador** actúa como el generador de datos interesantes. El publicador transmite esa información a los suscriptores a través del bróker. **El bróker** garantiza la seguridad verificando la autorización de los publicadores y los suscriptores.

MQTT representa un protocolo de mensajería ideal para el **IoT** y las comunicaciones **M2M** es capaz de proporcionar enrutamiento para dispositivos pequeños, económicos, de bajos recursos y con poca memoria en redes y baja ancho de banda

la siguiente figura muestra el proceso de **publicación/suscripción**

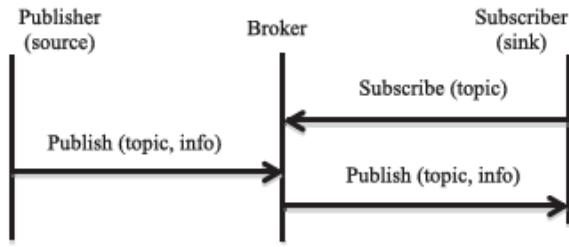
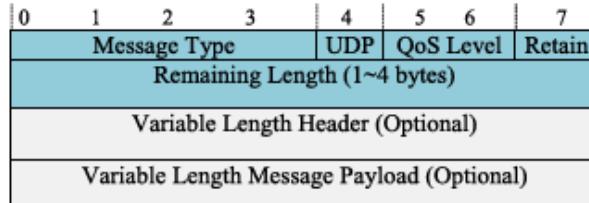


Fig. 9. Publish/subscribe process utilized by MQTT [70].

La siguiente muestra el formato de mensaje utilizado por **MQTT**



los primeros 2 bytes son el encabezado fijo. El valor del campo tipo de mensaje indica una variedad de mensajes: **CONNECT(1)**, **CONNACK(2)**, **PUBLISH(3)**, **SUSCRIBE(8)** y así sucesivamente. La bandera **DUP** indica que el mensaje puede estar duplicado. Hay 3 niveles de **QoS** para la garantía de entrega de mensajes. El campo retener indica al servidor que conservé el último mensaje recibido y lo envíe a los nuevos suscripciones como primer mensaje. El campo longitud restante, muestra la longitud restante del mensaje.

- 3) Extensible Messaging and Presence Protocol (XMPP): **XMPP** permite a los usuarios comunicarse entre sí enviando mensajes instantáneos por internet. **XMPP** permite que las aplicaciones de **IM** (mensajería instantánea) logren autenticación, control de acceso, medición de privacidad, encriptación de punto a punto e integral y compatibilidad con otros protocolos. En la siguiente imagen se ilustra su funcionamiento y en como los **gateways** pueden servir como puente entre redes de mensajería extranjera.

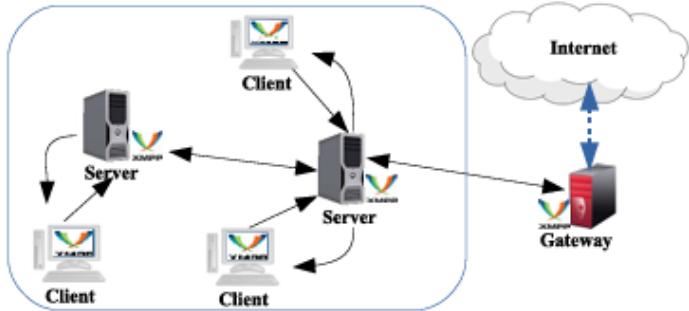


Fig. 11. Communications in XMPP.

XMPP es el preferido por la mayoría de las aplicaciones de mensajería instantánea, **XMPP** conecta un cliente a un servidor usando un flujo de fragmentos XML, un fragmento de XML representa un trozo de código que se divide en 3 componentes: **mensaje, presencia e iq (información/consulta)**. Las estrofas de mensaje identifican las direcciones origen y destino, los tipos y las ID que las entidades XMPP que utilizan un método de empuje para recuperar datos.

Un **bloque de mensajes** llenas los campos de asunto y cuerpo con el título y contenido del mensaje

El **bloque presencia** muestra y notifica a los clientes sobre las actualizaciones.

El **bloque iq** empareja remitentes y destinatarios de mensajes. La comunicación XML usando XMPP puede significar una sobrecarga a la red. Una solución sería comprimir los flujos XML utilizando **EXI**.

4) Advanced Message Queuing Protocol (AMQP):

Se centra en entornos orientados a mensajes. Soporta comunicación confiable de entrega de mensajes incluyendo entrega de mensajes incluyendo entrega como máximo una vez. AMQP requiere un protocolo de transporte confiable como TCP para intercambiar mensajes.

Las comunicaciones son manejadas por dos componentes principales: **intercambios y colas de mensajes**. Los intercambios se utilizan para enrutar los mensajes a las colas apropiadas. Este intercambio se basa en reglas y condiciones predefinidas. Los mensajes pueden almacenarse en las colas de mensajes y luego enviarse a los receptores.

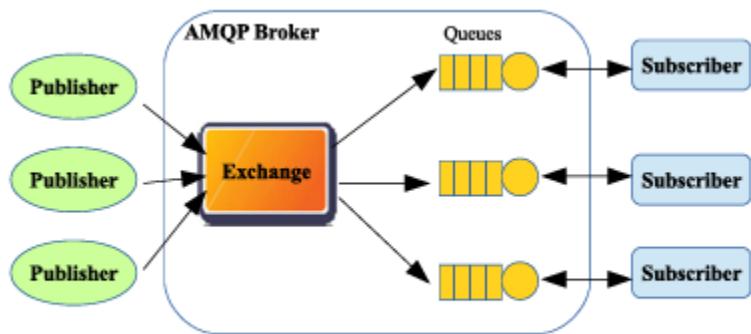
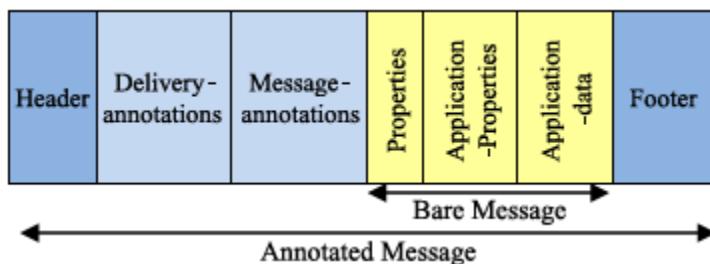


Fig. 13. Publish/subscribe mechanism of AMQP.

AMQP también admite el modelo de comunicaciones de **publicación/suscripción**.

las capacidades de mensajería se gestionan en la capa de transporte, **AMQP** define 2 tipos de mensajes: mensajes básicos que son proporcionados por el remitente y mensajes anotados que son vistos por el receptor.

veamos una imagen que ilustra el protocolo:



El encabezado transmite los parámetros de entrega, durabilidad, prioridad, tiempo de vida, primer adquirente y reencuentro de entregas.

En la capa de transporte las comunicaciones están orientadas a tramas en el protocolo **AMQP**, donde los primeros 4 bytes muestran el tamaño de la trama, **DOFF** (desplazamiento de datos). Indica la posición del cuerpo dentro de la trama. El campo tipo indica el formato y el propósito de la trama.

- 5) **Data Distribution Service (DDS)**: es un protocolo de **publicador/suscriptor** para la comunicación de M2M en tiempo real que se puede ver desplegado por un grupo de gestión de objetos. Este protocolo es una arquitectura sin intermediario utiliza multisésion para ofrecer una excelente calidad en sus servicios, esta arquitectura se adapta muy bien a las restricciones para las comunicaciones IoT y M2M. **DDS** soporta 23 políticas de QoS mediante las cuales el desarrollador puede abordar una variedad de criterios. La arquitectura se define en 2 capas: **publicación y suscripción centradas en datos (DCPS)** y **capa de reconstrucción local de datos (DLRL)**.

DCPS es responsable de entregar la información a los suscriptores. **DLRL** funciona como interfaz con las funcionalidades de **DCPS** la cual es opcional. 5 entidades son involucradas con el flujo de información en la capa de **DCPS**

- 1) Editor que difunde datos, 2) **DataWriter** es utilizado para interactuar con el publicador sobre los valores y cambios de datos. Esta asociación entre el punto 1 y el 2, indica que se va a publicar los datos especificados en un contexto proporcionado. 3) los suscriptores reciben la información de los publicadores y entregan a la aplicación. 4) **DatReader** utilizado por el suscriptor para acceder a la información recibida. 5) un tema que se identifica por un tipo de dato y un nombre. Los temas relacionan a los **DataWriter con los DataReader**.

La trasmisión de datos está permitida dentro de un dominio DDS, es un entorno virtual para aplicaciones conectadas de publicación y suscripción.

MQTT entrega mensajes con menos retardo que **CoAP** cuando la tasa de perdida de paquetes es baja. En cambio, cuando la tasa de perdida de paquetes es alta **CoAP** supera a **MQTT**.

CoAP consume mucha menos energía que **HTTP** gracias a su encabezado condensado y pequeño.

XMPP es una opción eficiente para aplicaciones web que requieren comunicación en tiempo real.

AMQP demostró mejores resultados que los servicios web **RESTful**

DDS es un protocolo que escala bien cuando se incrementan el número de nodos.

B. Protocolos de descubrimiento de servicios:

Los protocolos **DNS multicast (mDNS)** y **el descubrimiento de servicios DNS (DNS-SD)**, pueden describir recursos y servicios ofrecidos por dispositivos IoT

- 1) **DNS multicast (mDNS)**: es un servicio básico para la resolución de nombres. Este servicio puede realizar tareas de un servidor **DNS unicast**. Este servicio es flexible debido a que el espacio de nombres **DNS** se utiliza localmente sin gastos adicionales ni configuración. Es una opción ideal por las siguientes razones **a) no hay necesidad de una reconfirmación ni administración adicional, b) puede funcionar sin infraestructura, c) puede funcionar a pesar de fallas en la infraestructura**.

el servicio consulta nombres enviando un mensaje multicast IP a todos los nodos del dominio local. Con esto el cliente solicita que los dispositivos con el nombre dado respondan. Cuando la máquina objetivo recibe el mensaje envía un mensaje de respuesta por multicast que contiene su dirección IP. Como podemos ver en la imagen:

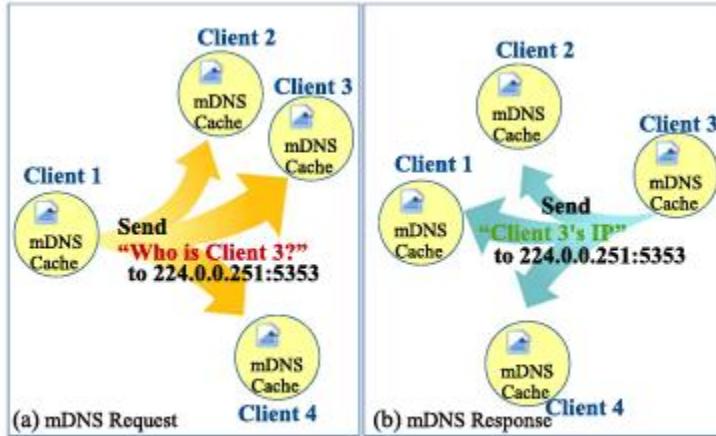


Fig. 17. Request/Response in mDNS protocol.

2) Descubrimiento de Servicios DNS (DNS-SD):

DNS-Based Service Discovery (DNS-SD) es un protocolo que permite a los clientes descubrir y emparejarse con servicios necesarios en una red local utilizando mDNS, eliminando la necesidad de configuración manual externa. Funciona enviando paquetes DNS vía UDP para localizar nombres de host estables y vincularlos con sus direcciones IP cambiantes, lo que garantiza la confiabilidad y permite que los dispositivos, como en el caso del IoT, entren y salgan de la red sin afectar el sistema. Aunque su principal desafío es el consumo de recursos por el almacenamiento en caché de entradas DNS, esto se soluciona limpiando el caché periódicamente, siendo Bonjour y Avahi las implementaciones más conocidas de esta tecnología.

C. Protocolos de infraestructura:

- 1) *Protocolo de enrutamiento para redes de baja potencia y alta pérdida (RPL):* El Protocolo de Enrutamiento para Redes de Baja Potencia y con Pérdidas (RPL), estandarizado por el grupo IETF ROLL, es una solución de enrutamiento independiente del enlace y basada en IPv6, diseñada específicamente para nodos con recursos limitados. Su objetivo principal es establecer una topología robusta capaz de soportar modelos de tráfico complejos (como multipunto a punto, punto a multipunto y punto a punto) incluso sobre enlaces inestables. La base estructural del RPL es el DODAG (Grafo Acíclico Dirigido Orientado al Destino), que organiza la red en un diagrama con un nodo raíz único; en esta estructura, cada nodo conoce a sus padres, pero no a sus hijos, asegurando siempre una ruta preferida hacia la raíz para optimizar el rendimiento.

Para construir la topología y mantenerla actualizada, RPL utiliza cuatro tipos de mensajes de control. El más crucial es el Objeto de Información del DODAG (DIO), iniciado por la raíz y propagado por la red para establecer el rango o nivel de cada nodo y seleccionar la ruta ascendente predeterminada (el parente preferido). Para el tráfico descendente (hacia los dispositivos), se utilizan mensajes DAO (Objeto de Anuncio de Destino), que informan a los padres sobre los destinos disponibles. Además, existen los mensajes DIS (para solicitar información de la red a nodos adyacentes) y DAO-ACK (para confirmar la recepción de los anuncios de destino).

El funcionamiento del protocolo contempla dos modos principales de operación: el modo "Non-Storing", donde el enrutamiento descendente se gestiona mediante enrutamiento desde la fuente (source routing), y el modo "Storing", donde los routers almacenan información y dirigen el tráfico basándose en las direcciones IPv6 de destino. Una implementación práctica común de este estándar en redes de sensores inalámbricos es ContikiRPL.

- 2) *6LowPAN:*

Las redes inalámbricas de área personal de baja potencia (WPAN), fundamentales para muchas comunicaciones IoT, presentan características restrictivas como un ancho de banda bajo y tamaños de paquete limitados (máximo 127 bytes en IEEE 802.15.4), lo que impide el uso directo de protocolos de capa de enlace anteriores. Para resolver esto, el grupo de trabajo IETF desarrolló el estándar 6LoWPAN, que actúa

como una capa de adaptación esencial para ajustar y mantener el tráfico IPv6 sobre estas redes restringidas.

Sus funciones principales son la compresión de encabezados para reducir drásticamente la sobrecarga de transmisión, la fragmentación para cumplir con los requisitos de la Unidad Máxima de Transmisión (MTU) de IPv6, y el reenvío a la capa de enlace para soportar la entrega de múltiples saltos. El protocolo organiza los datagramas utilizando combinaciones de encabezados identificados por dos bits: el encabezado de "Despacho" (01) se encarga de la compresión y multidifusión; el de "Direccionamiento de Malla" (10) identifica los paquetes que deben reenviarse; y el de "Fragmentación" (11) maneja los datos que exceden el tamaño de un cuadro (los marcados con 00 son descartados por no cumplir la especificación). Gracias a esta optimización, 6LoWPAN puede comprimir encabezados IPv6 hasta reducir su tamaño a solo dos bytes, permitiendo, en el mejor de los casos, enviar un datagrama IPv6 pequeño en un solo salto físico.

3) *IEEE 802.15.4:*

El estándar IEEE 802.15.4 especifica las capas físicas (PHY) y de control de acceso al medio (MAC) para redes inalámbricas de área personal de baja velocidad (LR-WPAN), convirtiéndose en la base fundamental para protocolos como ZigBee y aplicaciones IoT debido a su bajo consumo, bajo costo y capacidad para gestionar hasta 65,000 nodos con seguridad integrada, aunque sin garantías de QoS. Opera utilizando el método DSSS en tres bandas de frecuencia distintas, ofreciendo un equilibrio entre velocidad y alcance: 250 kbps en 2.4 GHz (para mayor rendimiento y baja latencia) y velocidades menores de 40 y 20 kbps en 915 MHz y 868 MHz respectivamente (para mayor sensibilidad y distancia), gestionando el tráfico y evitando colisiones mediante el protocolo CSMA/CA.

La arquitectura de la red se basa en dos tipos de nodos: los Dispositivos de Función Completa (FFD), que pueden actuar como coordinadores de la red (PAN coordinator), gestionar el enrutamiento y comunicarse libremente; y los Dispositivos de Función Reducida (RFD), nodos limitados que solo pueden conectarse con un coordinador. Esta flexibilidad permite implementar tres topologías principales: estrella, donde un coordinador central gestiona todo; punto a punto (malla), que permite la comunicación directa o escalonada entre nodos; y árbol de clústeres, una variación jerárquica de la anterior.

4) *Bluetooth de baja energía:*

Bluetooth Low Energy (BLE), o Bluetooth Smart, es una tecnología de radio diseñada para operar con un consumo energético mínimo, permitiendo que los dispositivos funcionen durante años con una potencia de transmisión entre 0.01 mW y 10 mW. Sus capacidades superan significativamente a las versiones clásicas, ofreciendo un rango de cobertura diez veces mayor (aproximadamente 100 metros) y una latencia 15 veces menor, además de ser más eficiente en términos de energía por bit que el protocolo ZigBee, lo que lo convierte en un estándar ideal para IoT y redes de sensores vehiculares.

La arquitectura de su pila de red comienza con la Capa Física (PHY) y la Capa de Enlace, encargadas de la transmisión de bits, acceso al medio y control de errores; sobre ellas, el protocolo L2CAP maneja la multiplexación y fragmentación de datos. En los niveles superiores, el perfil GATT optimiza la recolección de datos de sensores, mientras que GAP gestiona la configuración y los modos de operación (como publicidad o escaneo). BLE funciona bajo una topología de estrella con roles de maestro y esclavo: los esclavos emiten anuncios en canales dedicados para ser descubiertos por los maestros que escanean, permaneciendo la mayor parte del tiempo en modo de suspensión para conservar energía cuando no hay intercambio activo de datos.

5) *EPCglobal:*

EPCglobal es la organización encargada de gestionar y estandarizar el Código Electrónico de Producto (EPC), que funciona como una identificación única almacenada en etiquetas RFID para el rastreo en la cadena de suministro. Esta arquitectura es fundamental para el futuro del IoT debido a su apertura y escalabilidad. Los códigos EPC se clasifican principalmente en formatos de 64 y 96 bits; la versión de 96 bits es mucho más robusta, permitiendo identificar a más de 268 millones de empresas y miles de millones de productos individuales.

El sistema funciona mediante dos componentes físicos: la etiqueta (que contiene un chip con el ID y una antena) y el lector. La comunicación ocurre cuando el lector genera un campo de radiofrecuencia y la etiqueta refleja su señal devolviendo su número único. Posteriormente, el lector pasa este número al Servicio de Nombres de Objetos (ONS), una

aplicación que busca en una base de datos los detalles específicos del producto, como su fecha y lugar de fabricación.

La red completa de EPCglobal se compone de cinco partes: el código EPC, el sistema de identificación, el middleware, los servicios de descubrimiento (como el ONS) y los servicios de información. Una mejora significativa en esta tecnología fue la introducción de las etiquetas de Segunda Generación (Gen 2) en 2006, las cuales superaron a las etiquetas pasivas originales al ofrecer mayor interoperabilidad, fiabilidad y rendimiento a un costo más bajo.

6) *LTE-A (Evolución a Largo Plazo—Avanzada):*

LTE-Advanced (LTE-A) se posiciona como una solución celular óptima para infraestructuras de IoT y comunicaciones tipo máquina (MTC), especialmente en ciudades inteligentes, gracias a su durabilidad, escalabilidad y eficiencia en costos frente a otras opciones. A nivel físico, opera mediante el acceso múltiple por división de frecuencias ortogonales (OFDMA), dividiendo el ancho de banda en bloques de recursos físicos, y utiliza una técnica de portadoras componentes que permite agrupar hasta cinco bandas de 20 MHz para aumentar la capacidad.

Su arquitectura se sustenta en dos pilares: la Red Central (CN), encargada del control de dispositivos y flujos IP, y la Red de Acceso de Radio (RAN), compuesta por estaciones base (eNodeBs) interconectadas que gestionan la comunicación inalámbrica. Aunque los dispositivos pueden conectarse directamente a las estaciones o mediante pasarelas (gateways), el protocolo enfrenta desafíos significativos de congestión y calidad de servicio (QoS) cuando existe una saturación de accesos simultáneos; para mitigar esto, estudios indican que mantener los dispositivos MTC en estado inactivo por períodos más largos reduce la contención y mejora el rendimiento general de la red.

7) *Z-Wave:*

Z-Wave es un protocolo de comunicación inalámbrica de baja potencia diseñado específicamente para la automatización del hogar (domótica) y pequeños entornos comerciales, ideal para aplicaciones que requieren transmisiones de datos ligeras como el control de iluminación, seguridad, termostatos y sensores. Opera en bandas de frecuencia ISM alrededor de los 900 MHz, lo que le permite un alcance aproximado de 30 metros punto a punto con velocidades de transferencia que van desde los 40 kbps hasta los 200 kbps en sus versiones más recientes.

Su arquitectura técnica se organiza mediante nodos controladores y esclavos; el controlador gestiona la red completa manteniendo una tabla de la topología y utiliza el método de "enrutamiento de origen" (source routing), donde el dispositivo principal define la ruta exacta que debe seguir el paquete. Para garantizar la fiabilidad, su capa MAC incluye mecanismos de prevención de colisiones y la opción de enviar mensajes de confirmación (ACK) para asegurar que los datos llegaron correctamente.

D. Otros protocolos influyentes:

1) Seguridad:

Más allá de establecer la conectividad, el éxito del IoT depende críticamente de la seguridad y la interoperabilidad, un desafío complejo ya que los protocolos de seguridad tradicionales de Internet (diseñados para computadoras potentes) no sirven para dispositivos con recursos limitados. Por esto, la protección debe implementarse capa por capa con soluciones adaptadas.

A nivel de almacenamiento, existen soluciones como Codo para el sistema operativo Contiki, que cifra datos en el sistema de archivos de forma eficiente. En la transmisión, la seguridad se escala: la capa de enlace usa IEEE 802.15.4 para proteger conexiones entre vecinos, pero en la capa de red, IPsec (adaptado para 6LoWPAN) es superior para comunicaciones de múltiples saltos. En la capa de transporte, se estandariza el uso de TLS para TCP y DTLS para UDP. Finalmente, en la capa de aplicación, protocolos como CoAP utilizan versiones comprimidas como Lithe, mientras que MQTT, XMPP y AMQP se apoyan en TLS/SSL y autenticación SASL. Todo esto debe complementarse con Sistemas de Detección de Intrusos (IDS) para identificar ataques tanto desde la red interna como desde Internet.

2) Interoperabilidad (IEEE 1905.1):

Debido a la amplia variedad de tecnologías de red que utilizan los dispositivos IoT, es indispensable lograr la interoperabilidad entre ellas. Para solucionar esto en el ámbito de las redes domésticas digitales, se creó el estándar IEEE 1905.1, el cual funciona como una capa de abstracción que unifica y oculta la diversidad de las topologías de control de acceso al medio (MAC) sin necesidad de modificar las capas inferiores.

Gracias a este protocolo, distintas tecnologías de enlace y capa física — como Wi-Fi, Ethernet, IEEE 1901 (a través de líneas eléctricas) y MoCA (sobre cables coaxiales)— pueden coexistir e interactuar fluidamente bajo una misma interfaz. A pesar de este avance técnico que mejora la conectividad, el texto subraya que aún persisten preocupaciones significativas sobre el impacto ambiental de estos dispositivos y la necesidad de lograr despliegues a gran escala que sean ecológicamente sostenibles ("Green IoT").

VI. CRITERIOS DE CALIDAD DE SERVICIO, DESAFÍOS DEL IOT Y DIRECCIONES FUTURAS

Para que el Internet de las Cosas (IoT) funcione correctamente, es necesario resolver múltiples desafíos críticos como la disponibilidad, confiabilidad, movilidad, rendimiento, escalabilidad, interoperabilidad, seguridad, gestión y confianza. Superar estos obstáculos es fundamental para que los proveedores y programadores puedan implementar servicios eficientes.

1. Entre estos retos, destacan dos por su importancia global:
2. Seguridad y Privacidad: Debido a la sensibilidad de los datos de los usuarios.
3. Evaluación del Rendimiento: Medir qué tan bien funcionan los servicios es clave.

Existen varios proyectos de investigación (como IoT6, RERUM y RELYonIT) dedicados específicamente a investigar estas deficiencias y proponer soluciones, cuyos hallazgos suelen resumirse en encuestas académicas y tablas comparativas.

A. *Disponibilidad:*

La disponibilidad en el ecosistema IoT debe garantizarse simultáneamente en dos frentes: software, asegurando que los servicios estén accesibles para todos los usuarios en cualquier momento y lugar, y hardware, disponiendo de dispositivos siempre compatibles con protocolos esenciales como IPv6, 6LoWPAN, RPL y CoAP. Para lograr esta estabilidad operativa, la estrategia principal es la redundancia de componentes y servicios críticos, apoyada por el uso de herramientas de evaluación que permitan a los diseñadores medir y optimizar la disponibilidad desde las primeras fases del desarrollo del sistema.

B. *Fiabilidad:*

La confiabilidad en IoT se define como el funcionamiento correcto del sistema según sus especificaciones, lo cual garantiza la entrega exitosa de servicios y asegura la disponibilidad de la información a largo plazo. Este factor es crítico, especialmente en aplicaciones de emergencia, donde la red de comunicación debe ser totalmente resiliente a fallos.

Para evitar consecuencias desastrosas derivadas de retrasos, pérdida de datos o decisiones erróneas, la confiabilidad debe implementarse rigurosamente tanto en hardware como en software a través de todas las capas del sistema. Las investigaciones actuales proponen esquemas a nivel de transmisión para minimizar la pérdida de paquetes y utilizan modelos probabilísticos para evaluar el costo y la fiabilidad en la composición de los servicios.

C. *Movilidad:*

La movilidad es un desafío crucial en el IoT, ya que se debe garantizar la continuidad de los servicios mientras los usuarios y dispositivos se desplazan entre distintos puntos de acceso o gateways. Para evitar interrupciones y gestionar eficientemente la enorme cantidad de dispositivos, se han desarrollado diversas estrategias: desde esquemas de caching y tunneling que suplen la falta temporal de recursos, hasta la gestión de movilidad por grupos liderados por un nodo con patrones de movimiento similares. Además, se implementan soluciones específicas como la gestión distribuida del ciclo de vida de servicios para sensores, protocolos especializados para el Internet de los Vehículos (IoV) y mecanismos bioinspirados en el vuelo de aves para redes ad-hoc.

D. *Rendimiento:*

Evaluar el **rendimiento** de los servicios IoT es un desafío complejo, ya que depende de la eficacia de múltiples componentes y tecnologías subyacentes que deben monitorearse continuamente para ofrecer velocidad y costos accesibles. Aunque existen evaluaciones individuales para protocolos específicos como BLE, RFID, 6LoWPAN o RPL, el texto subraya que aún falta una metodología integral que permita medir el desempeño completo de las aplicaciones IoT, siendo este un problema abierto que requiere métricas que abarquen desde la velocidad de procesamiento hasta el factor de forma de los dispositivos.

E. Gestión:

La gestión de miles de millones de dispositivos IoT presenta un desafío crítico en las áreas de Fallos, Configuración, Contabilidad, Rendimiento y Seguridad (FCAPS), lo que ha impulsado el desarrollo de protocolos ligeros diseñados específicamente para evitar una pesadilla administrativa. Entre las soluciones más destacadas se encuentra el estándar **LWM2M** de la Open Mobile Alliance, que permite la administración remota de dispositivos M2M independientemente de la aplicación, y **NETCONF Light** del IETF, enfocado en la configuración de dispositivos con recursos limitados. Además de plataformas de monitoreo en tiempo real como MASH, se están implementando arquitecturas que delegan las tareas pesadas de gestión a los routers de borde o gateways para mantener la eficiencia y velocidad de la red.

F. Escalabilidad:

La escalabilidad en el IoT se define como la capacidad de incorporar nuevos dispositivos, servicios y funciones sin afectar negativamente la calidad de lo que ya está operando, un reto considerable dada la diversidad de hardware y protocolos de comunicación existentes. Para enfrentar esto, las aplicaciones deben diseñarse desde su base para ser extensibles. Existen propuestas de arquitectura que implementan "daemons" IoT divididos en capas (Objeto Virtual, Objeto Virtual Compuesto y Servicio), las cuales, mediante automatización y configuración cero, garantizan tanto la escalabilidad como la interoperabilidad. Además, se desarrollan plataformas PaaS para la entrega virtual de servicios y frameworks como IoT-iCore3, que buscan estandarizar mecanismos escalables para el registro, búsqueda y descubrimiento de entidades en la red.

G. Interoperabilidad:

La interoperabilidad de extremo a extremo es un reto crítico en el IoT debido a la necesidad de gestionar una inmensa cantidad de dispositivos heterogéneos y plataformas distintas. Para que los servicios lleguen a todos los usuarios sin importar su hardware (como ocurre con los smartphones que integran WiFi, NFC y GSM), tanto desarrolladores como fabricantes deben diseñar sistemas flexibles que permitan agregar nuevas funciones sin romper la integración existente. Un obstáculo importante es que diferentes proveedores pueden interpretar el mismo estándar de formas distintas, creando ambigüedades; para solucionar esto, son esenciales las pruebas de interoperabilidad en bancos de pruebas como los ETSI Plugtests y proyectos

de investigación como PROBE-IT, que validan la correcta comunicación entre protocolos como CoAP y 6LoWPAN.

H. Seguridad y privacidad:

La seguridad representa el desafío más complejo en la implementación del IoT debido a la falta de estándares y arquitecturas unificadas, lo que dificulta garantizar la privacidad en redes heterogéneas con miles de millones de dispositivos intercambiando información. Un problema crítico aún no resuelto en los estándares es la distribución de claves de cifrado entre los dispositivos (donde la arquitectura SOLACE del IETF está trabajando), además de la necesidad de un control de acceso granular que permita, por ejemplo, que un proveedor solo lea datos mientras otro tenga permiso para controlar el hardware; para esto se proponen soluciones como la segmentación en redes virtuales.

A pesar del gran interés gubernamental e industrial, la tecnología aún necesita madurar. Mientras que la seguridad y la arquitectura han recibido mucha atención, aspectos como la disponibilidad, confiabilidad y rendimiento siguen relegados mayoritariamente a simulaciones de laboratorio, faltando despliegues reales a gran escala para validarlos. Finalmente, una nueva corriente de investigación se enfoca en mejorar la localización de los objetos inteligentes para servicios sensibles al contexto, considerando reemplazar los métodos actuales basados en IP por nuevas infraestructuras como el Named Data Networking (NDN).

VII. ANÁLISIS DE BIG DATA, COMPUTACIÓN EN LA NUBE Y EN LA NEBLINA EN APOYO AL IOT

La conexión masiva de objetos físicos desde humanos y animales hasta smartphones y sensores genera volúmenes inmensos de información conocidos como Big Data, los cuales superan por mucho la capacidad de almacenamiento y procesamiento de los equipos de hardware y software tradicionales. Para solucionar esto, el Cloud Computing (computación en la nube) se presenta como la herramienta ideal; definido por el NIST como un modelo de acceso bajo demanda a recursos informáticos compartidos, permite gestionar redes, servidores y aplicaciones de forma remota, confiable y a bajo costo.

Dado que los dispositivos IoT (sensores y actuadores) producen datos constantes que requieren análisis complejos para extraer conocimiento útil, los recursos de la nube se convierten en la mejor opción para almacenar y procesar

esta información. De esta forma, se establece una relación simbiótica donde el IoT genera los datos y la nube proporciona la infraestructura necesaria para soportarlos, abriendo paso a conceptos como el Fog Computing para optimizar el análisis.

A. ***Analítica de Big Data en apoyo del IoT***

El valor real del Big Data para las empresas está en exprimirle análisis y conocimiento para ganar ventaja competitiva. Aunque existen plataformas como Apache Hadoop o SciDB, a veces se quedan cortas ante la inmensidad de datos del IoT, por lo que se requiere potencia en tiempo real (como lo hace Facebook) y aprovechar no solo los servidores, sino también la capacidad de cómputo de los propios dispositivos inteligentes para dividir el trabajo.

Lo ideal es dejar de hacer análisis específicos para cada aplicación y moverse hacia una plataforma común ofrecida como servicio. Un gran ejemplo es TSaaS (Time Series as a Service), que busca patrones en datos de sensores rapidísimo y ocupando poquísimo espacio. Finalmente, para no ahogarse en información, la estrategia es quedarse solo con los "datos interesantes" aplicando matemáticas avanzadas como el Análisis de Componentes Principales (PCA) o la reducción de dimensionalidad para filtrar el ruido.

B. ***Computación en la nube para el IoT***

El Cloud Computing es fundamental para procesar y extraer valor del Big Data generado por el IoT, aunque su implementación enfrenta retos significativos como la sincronización entre distintos proveedores, la falta de estandarización, el equilibrio de infraestructuras, y las diferencias críticas en seguridad y gestión entre los dispositivos y la nube. Para abordar esto existen múltiples plataformas como Google Cloud, Amazon y OpenIoT, destacando especialmente Xively y Nimbts. Xively es una solución PaaS (Plataforma como Servicio) que permite conectar dispositivos en tiempo real mediante protocolos como HTTP y MQTT, facilitando la visualización de datos y el control remoto de sensores con librerías listas para usar. Por su parte, Nimbts es una plataforma de código abierto que conecta sistemas embebidos a la nube para realizar análisis, generar alertas e interactuar con redes sociales utilizando XMPP. Al comparar estas opciones, aunque todas ofrecen interfaces para interactuar con sensores y lógica de negocio en la web, ninguna de las plataformas públicas actuales soporta el protocolo DDS.

C. Computación en la niebla en apoyo del IoT

El Fog Computing (o Edge Computing) funciona como el puente maestro entre tus dispositivos inteligentes y la nube masiva, acercando el procesamiento al "borde" de la red para que todo corra mucho más rápido. A diferencia de la nube tradicional, que es un monstruo de potencia centralizado pero lejano, el Fog son "micro" centros distribuidos —que incluso pueden estar en las torres de celular— diseñados para ofrecer menor latencia y permitir análisis en tiempo real.

Sus grandes ventajas son que reduce el lag, es mucho más barato de escalar (agregas nodos pequeños en vez de construir data centers gigantes) y hace un filtrado inteligente de datos on-the-fly antes de mandarlos a la nube central, ahorrando ancho de banda y mejorando la eficiencia. Aunque la tendencia actual es mover el análisis pesado al borde para dejar atrás los procesos lentos, todavía hay que "meterle galleta" a resolver temas de seguridad, movilidad y confiabilidad en estos dispositivos. Proyectos como RESERVOIR buscan que las nubes cooperen entre sí (nubes híbridas), apuntando a un futuro donde tienes la fuerza bruta del almacenamiento central combinada con la velocidad de respuesta local.

VIII. LA NECESIDAD DE UNA MEJOR INTEGRACIÓN HORIZONTAL ENTRE LOS PROTOCOLOS DE CAPA DE APLICACIÓN

Los dispositivos IoT se clasifican en dos categorías: dispositivos con recursos limitados y dispositivos con recursos abundantes. Los dispositivos con recursos abundantes pueden ejecutar TCP/IP y protocolos de aplicación como REST, CoAP, MQTT, AMQP, entre otros. En cambio, los dispositivos con recursos limitados no pueden ejecutar TCP/IP, lo que genera problemas de interoperabilidad. Además, incluso entre dispositivos que soportan TCP/IP, la diversidad de protocolos provoca fragmentación y limita el potencial del IoT.

Para abordar esta fragmentación, se han propuesto pasarelas de protocolo. Un ejemplo destacado es Ponte, desarrollado en el proyecto Eclipse IoT, que ofrece APIs unificadas para la conversión automática entre protocolos como CoAP y MQTT. Otros proyectos relacionados son Kura, Eclipse SCADA, Eclipse SmartHome y Krikkit, cada uno enfocado en aspectos como integración de dispositivos, visualización de datos, domótica o procesamiento en el borde.

Sin embargo, estas soluciones presentan limitaciones: asumen que los dispositivos subyacentes soportan TCP/IP, ignoran a los dispositivos con

recursos limitados y utilizan mensajes genéricos que generan ineficiencias y mayor verborrea en la comunicación. Además, el programador no tiene control sobre el protocolo de red subyacente.

Ante esta situación, se propone una nueva pasarela inteligente con las siguientes características:

- El programador debe tener control y flexibilidad para optimizar el protocolo de red según las necesidades de la aplicación.
- Los dispositivos con recursos limitados no deben ser tratados como ciudadanos de segunda clase.
- La pasarela debe ser oportunista y generar nuevas soluciones a partir de la fragmentación del mercado.

La solución propuesta consiste en una pasarela profundamente reprogramable mediante un lenguaje basado en reglas. Esta pasarela permite:

- Una pila de protocolos más ligera en dispositivos limitados, usando solo uIP/lwIP.
- Delegación de servicios de seguridad y gestión en la pasarela.
- Traducción de protocolos optimizable por el programador mediante reglas de alto nivel.
- Gestión autonómica local (FCAPS) sin intervención humana.
- Agregación de datos y flujos para reducir la carga en la red.
- Equilibrio de carga entre múltiples pasarelas.

Además, se introduce un mecanismo eficiente de traducción de protocolos basado en una tabla de índice de nombre-valor almacenada en cabeceras opcionales. Esto reduce la complejidad de conversión de $O(n^2)$ a $O(n)$, acelerando el proceso. Si no se usa la tabla, la conversión se realiza de forma convencional, siendo más lenta.

En conjunto, la propuesta complementa los esfuerzos de estandarización como IEEE 1905.1, LWM2M y NETCONF Light, ofreciendo una solución más inteligente, eficiente e inclusiva para la interoperabilidad en el IoT.

IX. CASOS DE USO DE APLICACIONES Y SERVICIOS

En esta sección se presentan tres casos de uso para ilustrar cómo los principales protocolos de IoT interactúan en aplicaciones reales, así como dos casos de uso adicionales centrados en capacidades de análisis de servicios. El enfoque está en la arquitectura general de las aplicaciones, no en fragmentos de código.

Los proyectos de código abierto disponibles en GitHub, desarrollados con Contiki/Cooja, incluyen:

- WSN: simulación de una red de sensores inalámbrica donde un nodo sink recibe paquetes para identificar el nodo ancla más cercano a cada nodo no ancla. Utiliza IPv6 y RPL.
- Service Discovery: implementación de un protocolo de descubrimiento de servicios en redes de sensores mediante multicast DNS (mDNS).
- Cloud Computing: envío de lecturas de temperatura desde múltiples motes a la nube Nimbots mediante HTTP REST.

Estos ejemplos son accesibles para entornos educativos y pueden ejecutarse tanto en simulación como en hardware real compatible con Contiki. Las guías de uso están disponibles en la wiki del proyecto.

A) Casos de uso de la aplicación

1) Sistema de Monitoreo de Pacientes en Hogares de Ancianos:

Las tres aplicaciones de IoT analizadas son: un sistema de monitoreo de pacientes en un geriátrico, un sistema para el control de trastornos alimenticios y un sistema de navegación en interiores para personas ciegas o con discapacidad visual. A continuación, se describe la arquitectura de la primera.

Para el monitoreo de pacientes en un geriátrico, se busca recolectar signos vitales y datos de sensores de luz y puerta para medir la actividad de los pacientes y detectar posibles casos de depresión. Una opción rápida es usar sensores SmartThings o BITalino con ZigBee o Z-wave, aprovechando sus APIs para enviar los datos a estaciones de enfermería.

Una alternativa personalizada consiste en usar sensores USB Phidgets con un microcontrolador o computadora de placa única (SBC). Estos nodos pueden comunicarse vía WiFi o IEEE 802.15.4. En este esquema,

se instala un broker MQTT como Mosquitto y se implementa un cliente MQTT con Eclipse Paho en el SBC para publicar los datos. Las estaciones de enfermería se suscriben al broker para recibir la información. Si se requiere colaboración entre sensores, puede usarse RPL para entrega de datos multisalto.

Para acceso remoto por parte de médicos, se puede desarrollar una aplicación móvil que se suscriba al broker MQTT. El broker puede exponerse en Internet mediante un firewall y una puerta de enlace M2M con conexión LTE-A.

2) *Monitoreo y Mitigación de los Trastornos Alimentarios:*

Se propone extender la aplicación para ayudar a pacientes con temblores esenciales o Parkinson a comer sin derramar alimentos. Para ello, se utiliza un guante equipado con sensores acelerómetros y pequeños motores MEMS vibratorios que contrarrestan la inestabilidad del movimiento. Debido a la necesidad de mínima latencia, se emplea el protocolo DDS para la comunicación directa entre sensores y actuadores, sin intermediarios.

Para integrar esta funcionalidad con las estaciones de enfermería, es necesario implementar una pasarela que traduzca los mensajes DDS a MQTT. La expansión de soluciones basadas en MQTT es sencilla, permitiendo añadir nuevos sensores y aplicaciones que publiquen o se suscriban al broker. Sin embargo, un único broker puede convertirse en un cuello de botella o punto de fallo ante un gran número de conexiones, por lo que se requiere el uso de múltiples brokers, lo que plantea retos interesantes en cuanto a topología y asignación de clientes.

3) *Sistema de navegación interior para personas ciegas y con discapacidad visual:*

Para extender la aplicación hacia un sistema de navegación en interiores para personas ciegas o con discapacidad visual, se emplea una constelación de transceptores decaWave o Nanotron que proporcionan servicios de localización en tiempo real (RTLS). Un dispositivo portado por el usuario utiliza mDNS para conectarse a un servidor local y obtener un token de autenticación. Los nodos RTLS pueden emplear DDS para el intercambio oportuno de datos y enviar la información recolectada al servidor local, que estima la posición del usuario y la superpone sobre un plano obtenido desde un servidor en Internet. De este modo, se genera

información táctil de navegación que permite evitar obstáculos y restricciones físicas reportadas previamente por otros usuarios. La Figura 28 integra visualmente los tres casos de uso y muestra cómo los protocolos de nivel de aplicación interactúan para lograr la funcionalidad completa del sistema IoT.

B) Casos de Uso Analíticos de Servicio

Además de la funcionalidad principal de las aplicaciones descritas, los desarrolladores pueden requerir la recolección de análisis de servicio y la gestión proactiva de los elementos que participan en la solución. A continuación, se presentan dos casos de uso orientados a este tipo de capacidades analíticas en despliegues IoT típicos.

1) Estimación Eficiente del Número de Direcciones IP Únicas que utilizan un Servicio Dado:

Para estimar de forma eficiente el número de direcciones IP únicas que acceden a un servicio IoT, se propone el uso de un filtro de Bloom invertible (IBF) en lugar de almacenarlas en una base de datos relacional, cuyo costo en memoria es elevado. En un experimento con flujos de $N=1000, 2000$ y 3000 paquetes, se evaluó la relación entre el tamaño del IBF y la precisión obtenida. Frente a una tabla relacional que requiere $4*N$ bytes, el IBF reduce significativamente el consumo de memoria. Por ejemplo, para $N=1000$ direcciones únicas con una precisión mínima del 95 %, el IBF utiliza solo un 56 % de la memoria del método tradicional, lo que demuestra su potencial para el análisis de servicios en entornos IoT a gran escala.

2) Seguimiento de la frecuencia de uso del servicio por una IP determinada:

Para optimizar la gestión de red y aplicaciones mediante el seguimiento de la frecuencia de uso de servicios IoT, se propone almacenar las direcciones IP y sus frecuencias de acceso como pares clave-valor en un filtro de Bloom invertible (IBF). En un experimento con 250 direcciones IP distintas que generaron 10,000 solicitudes de servicio, un IBF de solo 50 bytes permitió obtener el listado completo de IPs con sus frecuencias. En contraste, una estructura relacional habría requerido 1500 bytes (6 bytes por cada IP). Este resultado evidencia el importante ahorro de memoria que ofrece el IBF para el análisis de servicios en IoT, incluso en escenarios de mayor escala.

Conclusion:

El Internet de las Cosas (IoT) se consolida como una tecnología transformadora con un potencial de crecimiento exponencial, capaz de conectar miles de millones de dispositivos heterogéneos y generar un impacto profundo en sectores como la salud, el hogar, la industria, el transporte y las ciudades inteligentes. Sin embargo, su adopción masiva enfrenta desafíos estructurales que van desde la fragmentación de protocolos y la falta de estándares unificados, hasta limitaciones críticas en seguridad, interoperabilidad, gestión y eficiencia energética.

si bien existen numerosos protocolos de comunicación como CoAP, MQTT, XMPP, AMQP, DDS, RPL, 6LoWPAN, BLE, Z-Wave, entre otros y esfuerzos de estandarización por parte de organismos como IETF, IEEE u OMA, la diversidad de soluciones ha generado un ecosistema fragmentado. Esta fragmentación dificulta la integración horizontal entre dispositivos de diferentes capacidades y limita el desarrollo de aplicaciones verdaderamente escalables y universales.

Un hallazgo recurrente es la brecha existente entre los dispositivos con recursos abundantes, capaces de ejecutar TCP/IP y protocolos complejos, y aquellos con recursos limitados, que quedan relegados a un segundo plano en la mayoría de las propuestas actuales. Las pasarelas de protocolo tradicionales, como Ponte, ofrecen conversión automática, pero adolecen de rigidez, ineficiencia y nulo control por parte del programador, además de ignorar sistemáticamente a los dispositivos de bajos recursos.

Ante este panorama, se ha identificado la necesidad apremiante de una solución de integración más inteligente y flexible. La propuesta de una pasarela profundamente reprogramable, basada en un lenguaje de reglas de alto nivel, representa un avance significativo. Esta aproximación no solo coloca al programador en el centro del control del protocolo de red, permitiendo optimizaciones específicas por aplicación, sino que también reconoce a los dispositivos limitados como ciudadanos de primera clase dentro del ecosistema. Al delegar en la pasarela tareas de seguridad, gestión autonómica (FCAPS), agregación de datos y traducción eficiente de protocolos mediante mecanismos como la tabla de índice nombre-valor que reduce la complejidad computacional de $O(n^2)$ a $O(n)$ se logra una arquitectura más ligera, rápida y equitativa.

En paralelo, la gestión de la ingente cantidad de datos generados por el IoT exige el concurso de tecnologías complementarias. El Big Data, la computación en la nube y, especialmente, la computación en la niebla se perfila como aliados indispensables para procesar, filtrar y analizar información en tiempo real, reduciendo latencias y optimizando el uso del ancho de banda. No obstante, persisten retos en la estandarización de plataformas, la sincronización entre proveedores y la seguridad en entornos distribuidos.

Los casos de uso presentados monitoreo de pacientes, asistencia a personas con Parkinson, navegación para incidentes y análisis de servicios mediante filtros de Bloom invertibles demuestran la viabilidad práctica de combinar múltiples protocolos y arquitecturas para resolver problemas concretos.

el IoT se encuentra en una encrucijada: su crecimiento sostenido depende de superar la fragmentación actual mediante soluciones de interoperabilidad que sean a la vez inteligentes, eficientes e inclusivas. La pasarela reprogramable basada en reglas emerge como una respuesta sólida y complementaria a los esfuerzos de estandarización en curso. No obstante, el camino hacia un IoT verdaderamente ubicuo, seguro y sostenible aún requiere maduración en aspectos como la privacidad, la confiabilidad, la movilidad y el despliegue masivo en entornos reales. La investigación futura deberá orientarse no solo a perfeccionar los mecanismos de integración, sino también a garantizar que el IoT sea una tecnología al servicio de las personas, ética, ecológicamente responsable y accesible para todos.