A Layered Overview of Communication Architecture for IoT: Enabling the "Any Application–Any Device" Paradigm

Atef Bentahar[#1], Riad Saidi[*2], Tarek Bentahar[+3], and Abdelhakim Zeroual[4]

[#1]*Laboratory of Science for Mathematics, Computer Science and Engineering Applications*
Department of Computer Science, Institute of Science
University Center of Barika, Amdoukal Road, Barika, 05001, Algeria
[*]*LAAAS Laboratory*
Dept Commun Base Science and Technology
Mustapha Ben Boulaid Batna-2 University, Batna, Algeria
[+]*LABGET-laboratery*
Department of Electrical Engineering, Faculty of sciences and technology
Larbi Tebessi University, Tebessa, 12002, ALGERIA

[1]atef.bentahar@cu-barika.dz, [2]r.saidi@univ-batna2.dz, [3]tarek.bentahar@univ-tebessa.dz, [4]ab.zeroual@univ-batna2.dz

[1]https://orcid.org/0000-0003-3523-4459, [3]https://orcid.org/0000-0001-7976-9899, [4]https://orcid.org/0000-0002-4473-1492

Abstract— **The Internet of Things (IoT) paradigm, which interconnects billions of diverse, resource-constrained devices, requires specialized communication architectures to achieve massive scalability and interoperability. This paper provides a structured overview of the IoT communication stack from physical transmission mechanisms to service-level interactions, highlighting how traditional network models are adapted to meet IoT-specific constraints. Through comparative analysis, we underline the particular challenges posed by low-power and lossy networks, time-critical data processing, and the incorporation of lightweight, web-oriented service frameworks. A special focus is placed on the abstraction layer that enables the "Any Application–Any Device" paradigm through standardized application services. The study clarifies how these specialized architectures are effectively implemented in practice.**

Keywords— **IoT, Protocol Stack, IEEE 802.15.4, 6LoWPAN, RPL, CoAP, MQTT, AMQP, oneM2M, RESTful Services.**

I. Introduction

The Internet of Things (IoT) protocol stack can be viewed as an extension of the TCP/IP layered model. It comprises the following layers: physical, data link, internet, transport, application protocol, and application service (See Figure 1). Unlike the conventional TCP/IP model, the application layer is split into two layers in the IoT model: the application protocol layer and the application services layer [1].

This paper presents the IoT protocol stack according to the TCP/IP layering principles, with emphasis on the differences from TCP/IP and the distinctive characteristics of IoT systems.

II. Data Link Layer

The physical layer remains almost identical across TCP/IP, IoT, and OSI models. Therefore, the discussion begins with the data link layer, which supports open standards depending on the desired applications.

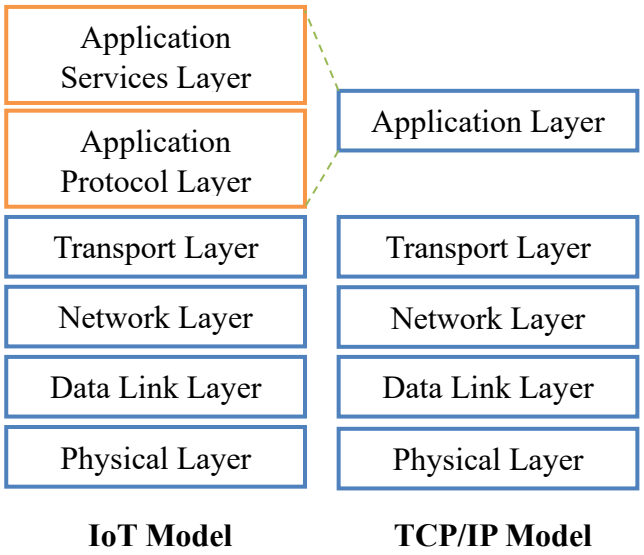

Fig. 1 IoT protocol stack

A. IEEE 802.15.4

IEEE 802.15.4, maintained by the IEEE 802.15 Task Group 4 (TG4) [2], defines the functioning of Low-Rate Wireless Personal Area Networks (LR-WPANs). It was designed for low-data-rate wireless connectivity with a focus on low complexity and long battery lifetimes, ranging from months to years. Initially focused on wearable devices, it has since expanded into applications such as sensors, interactive games, smart badges, remote controls, and home automation.

The standard specifies both the physical layer and the Medium Access Control (MAC) sub-layer, supporting multiple non-IP protocol stacks such as Zigbee, Zigbee Pro, WirelessHART, ISA 100.11a, and Routing Protocol for Low-Power and Lossy Networks (RPL) [3].

B. IEEE 802.15.4e TSCH

IEEE 802.15.4e [4] enhances its predecessor with improved energy efficiency and reliability. It introduces Time Synchronization and Channel Hopping (TSCH), which reduces power consumption and enhances reliability, while remaining compatible with existing 802.15.4 hardware.

C. IEEE 802.11ah (Wi-Fi HaLow)

Traditional Wi-Fi technologies are unsuitable for IoT due to high energy consumption and reliance on 2.4–5 GHz frequencies. IEEE 802.11ah (Wi-Fi HaLow) [5] operates below 1 GHz, enabling:

- Long-range transmission (up to 1 km outdoors)
- Support for up to 8191 devices per access point [6]
- Small, infrequent messages (~100 bytes, every >30s)
- Low data rates (~100 kbps)

The design philosophy prioritizes range and energy efficiency at the expense of throughput.

D. Time-Sensitive Networking (TSN)

Time-Sensitive Networking supports real-time IoT applications, particularly in industrial automation and automotive systems. Key standards include:

- IEEE 802.1Qca: explicit path control, redundancy, and IS-IS routing support.
- IEEE 802.1Qbv: deterministic scheduling with latency guarantees.
- IEEE 802.1CB: seamless redundancy by sending multiple packet copies along redundant paths.

III. Internet Layer

IoT deployments often operate in Low-power and Lossy Networks (LLNs), which are characterized by unreliable links, frequent topology changes, and energy constraints. Protocols defined by the IETF address these challenges: 6LoWPAN, RPL, and 6TiSCH.

A. 6LoWPAN

Protocol IPv6 for Low-Power Wireless Personal Area Networks (6LoWPAN) [Mulligan2007] adapts IPv6 to IEEE 802.15.4 by providing:

- Header compression (IPv6 headers reduced to as little as 2 bytes)
- Packet fragmentation and reassembly
- Mesh-under forwarding at Layer 2

This enables IPv6 communication in constrained LLNs.

B. RPL

RPL (Routing Protocol for Low-power and Lossy Networks) [7], standardized in RFC 6550, is a distance-vector protocol. It constructs a Destination-Oriented Directed Acyclic Graph (DODAG).

RPL supports:

- Objective functions (e.g., minimizing latency, maximizing delivery probability)
- Metrics (latency, reliability, energy)
- Constraints (e.g., avoiding battery-powered nodes or insecure links)

RPL is optimized for low memory consumption and adapts dynamically to changing topologies.

C. 6TiSCH

6TiSCH integrates IEEE 802.15.4e TSCH with IPv6 and RPL. It provides scheduling, deterministic multi-hop communication, and secure transmission, bridging real-time requirements with IoT needs.

IV. Application Protocols

Application protocols are the rules that manage communication between applications and various connected entities, such as devices (objects) or gateways.

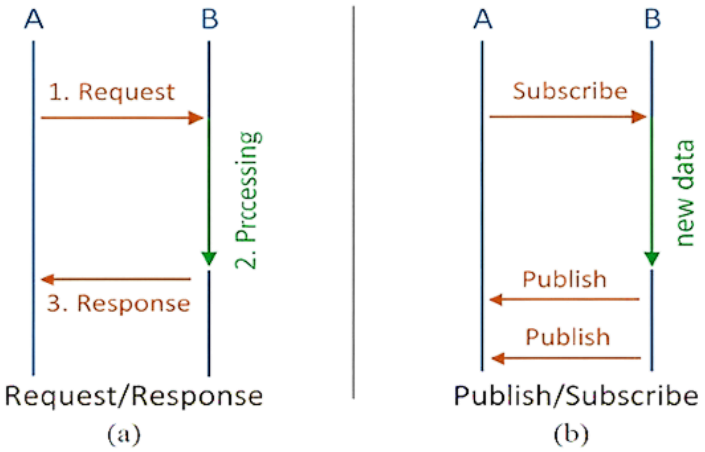

Fig. 2 The different interaction paradigms, (a) The Request/Response paradigm, (b) The Publish/Subscribe paradigm.

A. Communication Paradigms

Application protocols support two primary communication models for IoT interaction: Request/Response and Publish/Subscribe (Pub/Sub) as shown in Figure 2.

1) Request/Response Paradigm

Interaction: Allows for two-way communication where an initiator sends a request, and the recipient sends a response.

Suitability: Optimal for deployments following a client-server architecture or requiring interactive communication where both endpoints need to exchange information.

Limitation: Suboptimal for simple one-way data flow (e.g., a sensor sending data) due to unnecessary response message overhead.

2) Publish/Subscribe (Pub/Sub) Paradigm

Interaction: Enables one-way communication from a publisher (data producer, like a sensor) to one or more subscribers (data consumers, like an application).

Mechanism: Subscribers declare their interest in a specific data class or category, and the publisher sends messages when new data for that class is available.

Suitability: Optimal for IoT applications requiring one-way data flow.

Advantages: Offers loose coupling between communication endpoints and better scalability compared to client-server models.

Application protocols offer two modes for IoT messaging:

- Blocking: Sender waits for the operation to complete and for the final response.
- Non-Blocking: Sender receives a quick acknowledgment and retrieves the operation's result later.

B. RESTful Constraints

The six constraints that define a REST (Representational State Transfer) system are [8]: (I) Client/Server model, (II) The server must not store any client context that persists between requests (sessionless state). (III) Responses can be cached by clients and intermediate nodes. (IV) The system comprises a layered architecture that includes clients, servers, and potentially multiple intermediate nodes interspersed between them. (V) Client functionality can be extended or modified by the server via the transfer of executable code snippets that can be run on the client-side (e.g., scripts or applets). This is an optional REST constraint called "Code on Demand." And (VI) Uniform Interface. All interactions between clients and servers (or intermediate nodes) are governed by uniform interfaces. These interfaces use the notion of "resources." A resource is an abstraction for server-side information and the associated native data representation. Resources have unique identifiers (e.g., URIs in Web systems).

RESTful is the term used to describe any API or web service that implements the REST architectural style.

REST interfaces use a small, uniform set of operations (often called CRUD: Create, Read, Update, and Delete) that are applied universally to the representation of the resource.

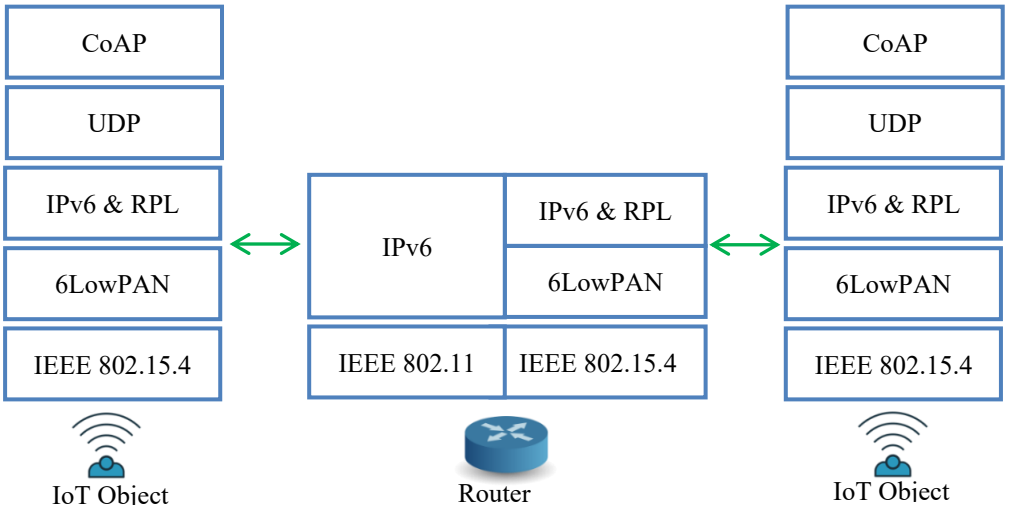

Fig. 3 Protocol stacks in object-to-object communication (CoAP example)

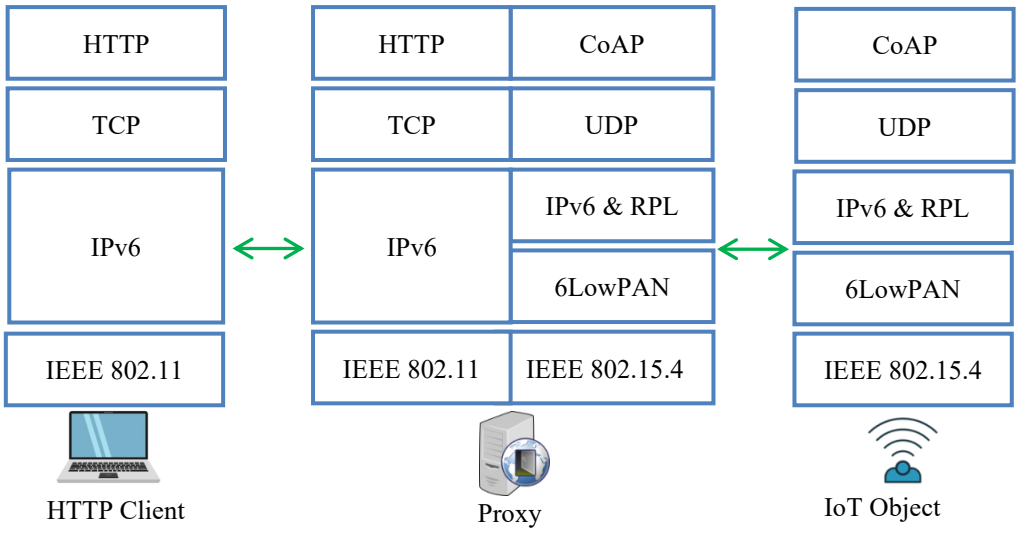

Fig. 4 Protocol stacks in human-to-object communication using HTTP proxy

C. IoT Application Protocols

In this section, we present the most interesting current IoT application protocols.

1) Constrained Application Protocol (CoAP)

CoAP [9], standardized by IETF CORE, is a lightweight RESTful protocol designed for constrained devices. It supports:

- CRUD operations
- Pub/Sub via the "Observe" option
- Resource discovery
- Optional reliable delivery and block transfers.

2) XMPP

XMPP [10], originally for instant messaging, is XML-based and extensible. It has been adapted for smart grids, network management, and IoT.

3) MQTT

MQTT [11], developed by IBM, is a lightweight Pub/Sub protocol.

- Clients connect to a broker.
- Topics are hierarchical and support wildcards.
- Designed for constrained devices with small bandwidth.
- Standardized by OASIS.

4) AMQP

AMQP [12] is a binary, message-oriented protocol. It supports:

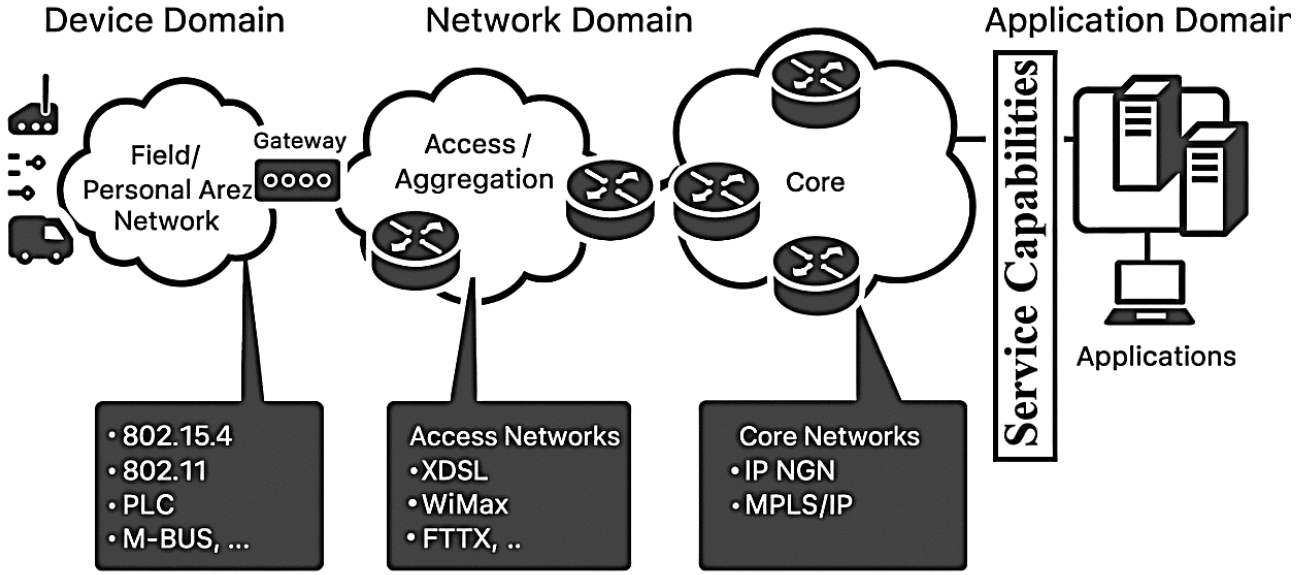

Fig. 5 ETSI M2M network architecture (Source [1]).

- Reliable delivery (at-most-once, at-least-once, exactly-once)
- Flow control with tokens
- Peer-to-peer or broker-based models
- Standardized by ISO/IEC and OASIS

5) SIP

Session Initiation Protocol (SIP) [13] sets up sessions for multimedia and IoT device communication, using proxies for routing, authentication, and authorization.

6) IEEE 1888

IEEE 1888 [14] supports environmental monitoring and smart energy. It uses XML/SOAP (Simple Object Access Protocol) for time-series data exchange, with identifiers managed through URIs.

7) DDS RTPS

DDS-RTPS [15] is a data-centric real-time Pub/Sub protocol. It supports:

- Automatic discovery of publishers/subscribers
- QoS policies (reliability, persistence, timeliness)
- UDP transport with multicast or brokers.

Figures 3 and 4 illustrate the two primary interaction models: object-to-object and PC-to-object communication. These models are designed for resource-constrained IoT objects and span the entire protocol stack up to the application layer.

V. Application Service Layer

The service layer abstracts device heterogeneity, enabling the paradigm of "any application – any device." It reduces complexity and enhances interoperability.

i. The Problem (Current Shortcoming of IoT):

- Application code is tied to the specifics of individual devices.
- Replacing a device (even with one from a different manufacturer) or adding a new device type requires modifying the application's source code, leading to high development and maintenance costs.
- Device networks are often closed and proprietary, requiring complex and costly application gateways for interconnection.

ii. The Solution (The Abstraction Layer):

- An Abstraction Layer is needed between applications and devices (objects).
- This layer provides a common set of services that allows an application to interface with any device without needing to understand its specific internal elements.
- It enables the "any application - any device" paradigm and is called the Application Services layer in the IoT protocol model.

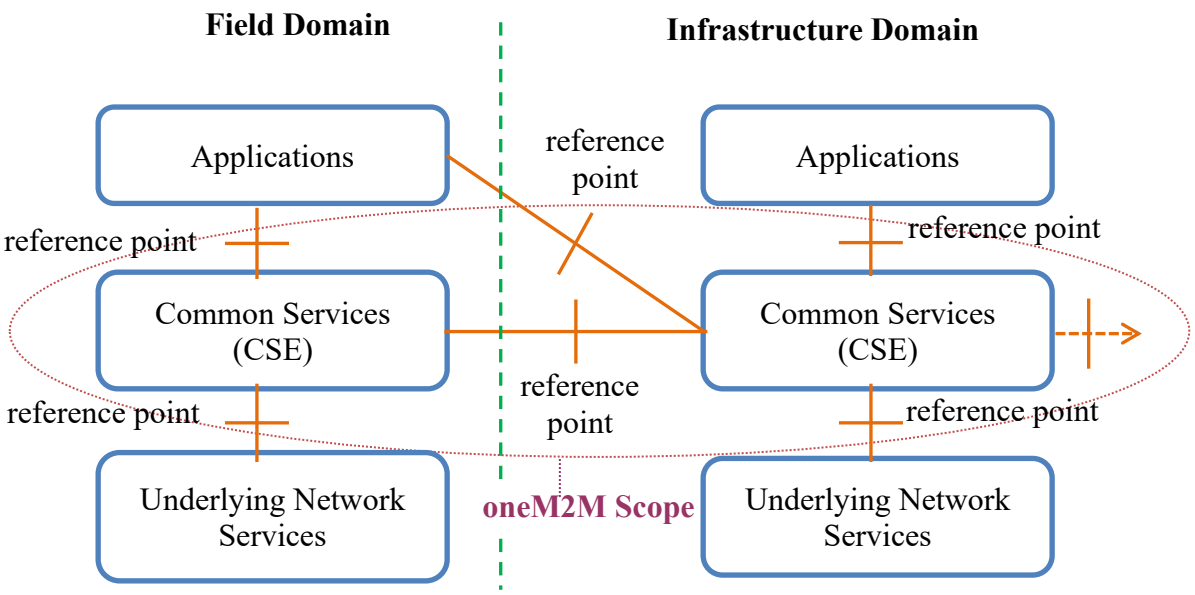

Fig. 6 OneM2M network architecture showing reference points.

- Its function is to provide transparent interoperability and promote agile development of IoT solutions.

iii. Business Driver (Communication Service Providers - CSPs):

- CSPs are driving this layer to derive additional revenue from their networks by differentiating beyond simple IP connectivity.
- They recognize the value of IoT is in the data, not its transport.
- The Application Services layer aims to transform the network into a common platform for various IoT applications.

iv. Standardization Efforts:

- The European Telecommunications Standards Institute (ETSI) published the first version of its Machine-to-Machine (M2M) service layer standard in 2012 [16], which includes Thing-to-Thing communications and defines a standardized platform.
- In the same year, a global organization called oneM2M [17] was launched by seven standards development organizations to jointly define and standardize the common horizontal functions of this Application Services layer.

A. ETSI M2M

In this section, we describe the architecture and functionality of the ETSI M2M standard's Service Capabilities Layer (SCL).

1) ETSI M2M Architecture Domains:

The network architecture is divided into three domains:

Device Domain: Contains physical devices (entities that provide or transmit data) and gateways (which ensure inter-operation between non-IP terminals and the network).

Network Domain: Includes the communication networks that interconnect gateways and applications.

Application Domain: Contains vertical-specific applications (e.g., smart city, e-health) and the Service Capabilities Layer (SCL).

2) Service Capabilities Layer (SCL):

Function: The SCL is a Middleware layer that provides common functions to various applications, simplifying development and deployment by hiding network specifics.

Location Types: SCL functions can reside in different locations, leading to three types:

- D-SCL: Device SCL (on devices).
- G-SCL: Gateway SCL (on gateways).
- N-SCL: Network SCL (on servers/data centers).

Hierarchy: The ETSI M2M model uses a strict two-level hierarchy, with the N-SCL at the top and D-SCL/G-SCL at the lower level; deeper daisy-chaining is not supported.

Key SCL Functions: The SCL provides essential common services, including [1]:

- Registration of devices, applications, and remote SCLs.
- Synchronous and asynchronous data transfer.
- Identification of applications and devices.
- Group management for mass operations.
- Security mechanisms (authentication, authorization, access control).
- Remote device management and location information.
- Location information

Architectural Style: ETSI M2M uses a RESTful architectural style, where all SCL data (including device-generated data, device information, and access rights) is represented as resources and is uniquely addressable via URI.

The architecture of ETSI M2M is summarized in Figure 5.

B. oneM2M

The oneM2M standard conceptualizes any IoT deployment within two primary domains (See Figure 6) :

Field Domain: Encompasses physical objects (devices) and their associated gateways.

Infrastructure Domain: Includes communication networks (for aggregation and core connectivity) and data centers.

Functionally, each domain integrates three core entity types:

Application Entity (AE): Implements the specific application logic (e.g., home automation, smart parking) and can reside on various physical nodes.

Common Services Entity (CSE): This middleware layer acts as an abstraction between AEs and underlying network services. Logically equivalent to ETSI M2M's SCL, the CSE provides a standardized set of common functions to applications, including:

- Identity and Registration Management: For AEs and CSEs.
- Connectivity Management: Ensures efficient and scalable network utilization.
- Remote Device Management: For configuration and diagnostics.
- Data Exchange: Facilitates storage, sharing, and event notifications between applications and devices.
- Security and Access Control: Manages access permissions to data.
- Discovery: Enables the location of entities, data, and resources.
- Group Management: Supports bulk operations.
- Location Services: Offers abstracted location information.

Network Services Entity (NSE): Delivers value-added network services to the CSE, such as Quality of Service (QoS), advanced device management, and device triggering.

oneM2M defines specific reference points (interfaces) for communication between these entities, including a crucial interface between the CSE and NSE [18]. This interface allows the CSE (service layer) to access and request network services (e.g., signaling QoS needs, transmission scheduling, device triggering, or retrieving location data from the transport layer).

VI. Conclusions

This paper has provided a comprehensive, layered overview of the Internet of Things (IoT) protocol stack, moving from the foundational data link layers to the application service layer. This examination underscored the necessity for specialized adaptations of traditional TCP/IP models to address the unique challenges of connecting billions of heterogeneous, resource-constrained devices.

A critical finding is the imperative need for an abstraction layer (such as ETSI M2M SCL or oneM2M CSE). This middleware crucially decouples applications from device specifics and underlying network services, offering standardized functions like identity management, data exchange, and security, thereby simplifying development and enabling transparent interoperability within the application layer. This functional abstraction is key to achieving the "Any Application–Any Device" paradigm, fostering agile development and scalability. Strategically, this service layer empowers Communication Service Providers to monetize IoT data by creating common, data-centric platforms, ultimately enabling the ecosystem's vast scale, manageability, and commercial viability.

Looking ahead, the evolution of the IoT landscape will be defined by the convergence of Edge Computing and Artificial Intelligence (AIoT), which

will push intelligence closer to the data source. While current abstraction layers provide essential interoperability, future research must address the complexities of decentralized governance and the stringent latency requirements of real-time, autonomous systems. As we transition toward 6G and satellite-integrated networks, the challenge will shift from simple connectivity to achieving semantic interoperability and energy-neutral operation. These emerging trends ensure that the protocol adaptations and architectural frameworks discussed herein will remain a foundational prerequisite for a secure, intelligent, and sustainable global ecosystem.

REFERENCES

[1] A. Rayes and S. Salam, "IoT Protocol Stack: A Layered View," Internet of Things From Hype to Reality. Springer International Publishing, pp. 93–138, Oct. 23, 2016. doi: 10.1007/978-3-319-44860-2_5.

[2] IEEE 802.15 WPAN™ Task Group 4 (TG4). (March 2004). http://www.ieee802.org/15/pub/TG4.html.

[3] Howitt and J. A. Gutierrez, "IEEE 802.15.4 low rate - wireless personal area network coexistence issues," 2003 IEEE Wireless Communications and Networking, 2003. WCNC 2003., vol. 3. IEEE, pp. 1481–1486. doi: 10.1109/wcnc.2003.1200605.

[4] R. C. A. Alves and C. B. Margi, "IEEE 802.15.4e TSCH Mode Performance Analysis," 2016 IEEE 13th International Conference on Mobile Ad Hoc and Sensor Systems (MASS). IEEE, Oct. 2016. doi: 10.1109/mass.2016.054.

[5] IEEE Standard for Information technology--Telecommunications and information exchange between systems - Local and metropolitan area networks--Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 2: Sub 1 GHz License Exempt Operation. doi: 10.1109/ieeestd.2017.7920364.

[6] T. Y. Arif, R. Munadi, S. Syahrial, and M. M. Rizky, "Performance of Centralized Authentication Control in IEEE 802.11ah with Multirate Registration," 2019 International Conference on Electrical Engineering and Computer Science (ICECOS). IEEE, pp. 265–270, Oct. 2019. doi: 10.1109/icecos47637.2019.8984517.

[7] T. Tsvetkov, "RPL: IPv6 Routing Protocol for LOW Power and Lossy Networks," Chair for Operating Systems and Chair for Network Architectures and Services, Department of Computer Science, Technische Universität München, 2011, doi: 10.2313/NET-2011-07-1_09.

[8] T. Erl, R. Balasubramanian, C. Pautasso, H. Wilhelmsen, B. Carlyle, and D. R. Booth, "SOA with REST: Principles, patterns & constraints for building enterprise solutions with REST," Upper Saddle River, NJ: Prentice Hall, 2012, ISBN 978-0-13-701251-0.

[9] K. Hartke and M. Richardson, "Extended Tokens and Stateless Clients in the Constrained Application Protocol (CoAP)," RFC Editor, Jan. 2021. doi: 10.17487/rfc8974.

[10] F. Baker and D. Meyer, "Internet Protocols for the Smart Grid," RFC Editor, June 2011. doi: 10.17487/rfc6272.

[11] MQTT Version 5.0. Edited by Andrew Banks, Ed Briggs, Ken Borgendale, and Rahul Gupta. 07 March 2019. OASIS Standard. https://docs.oasis-open.org/mqtt/mqtt/v5.0/os/mqtt-v5.0-os.html. Latest version: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html.

[12] J. O'Hara, "Toward a Commodity Enterprise Middleware," Queue, vol. 5, no. 4, pp. 48–55, May 2007, doi: 10.1145/1255421.1255424.

[13] J. Rosenberg et al., "SIP: Session Initiation Protocol," RFC Editor, June 2002. doi: 10.17487/rfc3261.

[14] IEEE 1888-2014 - IEEE standard for ubiquitous green community control network protocol. (2014). Ieee.Org. Retrieved October 21, 2021, from https://standards.ieee.org/standard/1888-2014.html.

[15] About the DDS interoperability wire protocol specification version 2.5. (June 2021). Omg.Org. Retrieved October 21, 2021, from https://www.omg.org/spec/DDSI-RTPS.

[16] ETSI completes foundation standards package for M2M services. (20 February 2012). Etsi.Org. Retrieved October 21, 2021, from https://www.etsi.org/newsroom/news/387-news-release-20-february-2012.

[17] OneM2M The IoT Standard. Homepage https://www.onem2m.org/

[18] Martigne, P. (2015). Overview of ETSI machine-to-machine and oneM2M architectures. In Machine-to-machine (M2M) Communications (pp. 27–46). Elsevier. https://doi.org/10.1016/b978-1-78242-102-3.00002-2.