

Communication Interfaces, Protocols, and Specialized Time Series Databases for the Internet of Things (IoT) - Survey

Georgi Petrov, Velimir Vasilev

The aim of this review is to systematize the main communication interfaces, protocols, and specialized databases for storing temporal information and records used for storing data and signals from IoT, thereby facilitating new IoT application developers. The review examines classical methods of communication for connected devices in the industry and the evolution of communication protocols and interfaces that support the widespread adoption of IoT today.

Keywords – IoT, protocols and database.

Комуникационни интерфейси, протоколи и специализирани бази данни с времеви серии за Интернет на нещата (IoT) - обзор (Георги Петров, Велимир Василев) Целта на обзора е да систематизира основни комуникационни интерфейси, протоколи и специализирани бази данни за съхраняване на темпорална информация и записи, използвани за съхраняване на данни и сигнали от IoT, като така да улесни разработчиците на нови IoT приложения. Разгледани са класическите методи за комуникация на свързани устройства в индустрията и еволюцията на комуникационните протоколи и интерфейси подпомагащи масовото внедряване на IoT.

Introduction

For many years, networked sensors and actuators were an area of interest primarily for the industrial sector. Systems for building, industrial, and transportation automation were generally territorially fixed, and their integration relied on SCADA (Supervisory Control and Data Acquisition) [1]. These systems use various communication protocols between control devices such as PLCs (Programmable Logic Controllers) and RTUs (Remote Terminal Units) [1], [2], as well as different sensors, and for connection with servers that collect and process data. They are widely used in monitoring and managing production lines, the energy sector, and water supply systems. Their relatively high cost and expensive maintenance and development made them inaccessible to small and medium-sized enterprises for many years.

With the emergence of the Internet of Things (IoT), SCADA systems gained new capabilities for enhanced data collection and processing, process optimization, and even automation of certain actions within the devices themselves, later known as "edge computing." Significant progress was made following the production of the first Ethernet chip – Intel 82586 (1982), followed by National Semiconductor DP8390 (1985) and Realtek RTL8010 (1989) [3]. The

introduction of Ethernet enabled high-speed data transmission and significantly higher performance compared to previous communication protocols based on twisted pair and standards like RS-432, RS-485, etc. (see Table 1). This allowed the use of newly emerging, affordable network switches and applications.

Table 1
Commonly used interfaces in industrial applications

Interface	Maximum Number of Devices	Maximum Length	Maximum Speed (Periphery)
RS-232 (1960s)	1 (point-to-point)	15 m	up to 115,200 bps
RS-422 (1970s)	10 (point-to-multipoint)	1200 m	up to 10 Mbps
RS-485 (1983)	32 (point-to-multipoint)	1200 m	up to 10 Mbps
Modbus RTU (1979)	32 (point-to-multipoint)	1200 m	115,200 bps (depending on settings)
HART (1980s)	15	2000 m (two-wire cable)	1200 bps (at 4-20 mA)
Interbus (1987)	256	500 m to 1.5 km	2 Mbps (max.)

Interface	Maximum Number of Devices	Maximum Length	Maximum Speed (Periphery)
CAN (1986)	110	40 m (at 1 Mbps); up to 1 km (at 125 kbps)	1 Mbps (max.)
Profibus (1989)	126	100 m to 1,000 m	12 Mbps (max.)
Ethernet (1980s)	Thousands	100 m (Category 5 cable)	100 Mbps, 1 Gbps, 10 Gbps
Modbus TCP/IP (1999)	256	100 m	100 Mbps (via Ethernet)
EtherCAT (2003)	Up to 65,535	100 m (Category 5 cable)	100 Mbps (max.)

With these specialized chipsets, Ethernet connectivity was integrated as an extension of traditional serial communication interfaces through a process known as "serial port virtualization" or via ready-made Serial-to-Ethernet converters. In Windows OS, specialized software and drivers are used, while in Linux OS, standard relay tools such as "Socket CAT" can be configured to transfer data between a TCP/IP socket and a serial port. These system utilities allow the connection of various types of I/O (input-output), such as file descriptors, network sockets, serial ports, and many others, without requiring a dedicated converter.

Microcontrollers with built-in Ethernet/WiFi

One of the first microcontrollers with built-in TCP/IP and Ethernet support was released in the early 1990s, the NS32 8032 Ethernet interface from National Semiconductor, but due to the lack of a well-developed Internet connection, it remained less popular. In 2007, Microchip released a variation of the PIC32 using the ENC28J60 Ethernet controller. This was a significant milestone in the direct integration of Ethernet communication into microcontrollers. In terms of Wi-Fi, Espressif Systems [4] pioneered this technology in 2014 with the ESP8266 microcontroller. For a long time, it was the only one and widely popular in IoT devices and prototypes. Its successor, the ESP32, was released in 2016 with a more powerful processor and Bluetooth 5.0 support, ideal for home and office IoT applications. It was this chip that made it possible to create a huge number and brands of IoT devices [5]. ESP8266 supports WiFi 802.11b: up to 11 Mbps, 802.11g: up to 54 Mbps,

802.11n: up to 72.2 Mbps (maximum), considering that the maximum size of the RX/TX buffer is 2x128 bytes, which is a common limiter when implementing streaming applications. It should be borne in mind that IoT can use different radio communication modules and protocols (ZigBee, LoRa, MicrelNET) (see Table 2), which usually require a gateway for their connection to the Internet [6]. In addition, firmware for integrated control of these systems has also been developed. For example, integration with Tasmota and Node-RED provides powerful capabilities for control and automation of IoT devices. Tasmota is a popular firmware platform for ESP8266/ESP32-based devices, which allows easy connection and control of IoT devices over local networks and servers. It offers support for various communication protocols such as MQTT, HTTP, and CoAP. Node-RED is an open source tool that allows graphical programming for connecting devices, services, and APIs. It is an easy way to create automated workflows that can interact with Tasmota-powered devices [7] using MQTT, HTTP, or other protocols.

Table 2
Wireless technologies used in IoT

Protocol	Minimum Speed	Maximum Speed	Minimum Distance	Maximum Distance
LoWPAN	Low	Medium	Very Small	Medium
LoRaWAN	Very Low	50-100 kbps	2-3 km	Up to 15-20 km
Zigbee	Low	250 kbps	10-20 m	Up to 100 m
Bluetooth Low Energy (BLE)	1 Mbps	2 Mbps	1-10 m	Up to 100 m
4G	100 Mbps	1 Gbps	1-2 km	Up to 10 km
5G	100 Mbps	10 Gbps	1-10 km	Up to 100 km
NB-IoT	20-250 kbps	250 kbps	Small	Up to 10 km

IoT-specific protocols

The protocols used in IoT fall into one of three categories, see Table 2:

- packet-oriented (IP)
- message-based
- notification-based (MQTT)

In packet-oriented protocols, the order of reception may differ from the order of transmission, which requires reordering and fragmentation, but allows large volumes of data to be reliably sent and received. In packet-oriented networks, the order of reception may differ from the order of transmission, which requires reordering and fragmentation, but allows large volumes of data to be reliably sent and received. Message-based protocols organize data as a single message, such as SMS (Short Message Service) over the GSM network, which contains structured information (e.g. header, metadata, and a payload of limited size, e.g. 160 bytes) [8]. They typically use "publish/subscribe" or "request/response" mechanisms. Suitable for IoT, chat, automation, and other real-time systems.

Notification-based protocols rely on event-triggered notifications, where a device or service sends a short message to clients (e.g., a server or another device). It is usually asynchronous - the recipient can receive the message without sending a request. Examples are: WebSockets – supports bidirectional connections for real-time notifications, HTTP/2 Server Push – allows the server to send data to the client without an explicit request. Push notifications (e.g. Firebase Cloud Messaging, Apple Push Notification Service).

Direct IP support capabilities allow IoT to communicate via TCP/UDP (Transmission Control Protocol/User Datagram Protocol)-based protocols. This makes them accessible for building devices that operate both in on-premises applications, through a hub or a dedicated cloud for access. One of the basic protocols for user access is HTTP (Hypertext Transfer Protocol) [9], which is designed for transmitting hypertext on the WEB (World Wide Web). In IoT, it can be used to exchange data with web servers or cloud platforms (see Table 3). Unfortunately, it is not the most efficient protocol for IoT devices requiring high energy efficiency, it uses a wider bandwidth, but it is essential for creating interfaces for initial device initialization by individual users. It remains the most universal and standardized way for a device to make database entries, verify information on websites, and perform logins based on a username and password. HTTPS can also be used for improved security, but it requires more powerful controllers.

CoAP (Constrained Application Protocol) is an application protocol designed specifically for IoT. It is based on UDP and designed for closed and managed

networks. It is lightweight and energy efficient, providing significantly lower message overhead compared to HTTP. It is based on UDP, which makes it fast, but has limited compatibility with traditional web technologies and requires a managed infrastructure as there is no direct quality management.

MQTT (Message Queuing Telemetry Transport) is probably the most popular real-time messaging protocol. It is a lightweight and efficient protocol designed for resource- and bandwidth-constrained situations, with speed and reliability as a priority. It works through a "publish/subscribe" model, has low overhead, and is suitable for sensors and devices with small memory. It is not best suited for large packets, files, or high-bandwidth applications, such as streaming real-time sensor data and images.

Table 3

IoT data transfer protocols

Protocol	Data Volume	Security	Speed	Real Time	Based On
HTTP	High	Medium	Medium	No	Packets
HTTPS	High	High	Medium	No	Packets
CoAP	Low	Medium	High	Yes	Messages
MQTT	Medium	Medium	High	Yes	Packets, Messages
XMPP	Medium	High	Medium	Yes	Messages
LWM2M	Low	High	Medium	Yes	Messages
WebSockets	Medium	Medium	High	Yes	Packets

XMPP (Extensible Messaging and Presence Protocol) is a specialized protocol for real-time communication. It is mainly used for chat and messaging. It has also been extended to IoT applications, where reliable and secure communication between devices that have intensive mesh connectivity is needed. It is more complex than the ones described above, but has good security.

LWM2M (Lightweight Machine-to-Machine) is intended for device management and monitoring. Designed for IoT devices with limited resources. It provides a means of managing and monitoring devices, using CoAP or SMS for communication. Suitable for mobile devices but requires a specific server for support. A major problem with IoT remains IPv6 support, for a similar solution, the LoWPAN protocol (IPv6 over Low Power Wireless Personal

Area Networks) was created, see Table 3. This is a standard that allows sending IPv6 packets over low-bandwidth networks, such as Zigbee, and is designed for IoT applications with limited resources. NAT64 (Network Address Translation 64) is another technology that enables communication between IPv6 and IPv4 networks by translating addresses and packets between the two protocols, and allowing seamless communication between IPv6 and IPv4 devices.

LoRaWAN (Long Range Wide Area Network) is suitable for long-distance communication with low power and low speed. Designed for applications requiring long-distance communication with minimal power consumption. It uses specific network topologies that are well suited for IoT devices with low bandwidth. It uses Chirp Spread Spectrum (CSS) modulation and operates at 868 MHz (for Europe), 915 MHz (for North America), and 433 MHz (for Asia), and is available in other frequency bands depending on the region of use. The lower frequencies make it suitable for rugged terrain and buildings.

Zigbee is a popular wireless protocol for low-power devices based on IEEE 802.15.4 for the physical layer [10]. This makes it a suitable choice for building and industrial automation systems. It uses BPSK (Binary Phase Shift Keying) and O-QPSK (Offset Quadrature Phase Shift Keying) modulation, and modules are available in the following frequency ranges: 2.4 GHz (global), 868 MHz (for Europe), and 915 MHz (for North America), there are modified devices with quite high emission levels around 0 dBm to +8 dBm, higher powers are also possible, which can reach +20 dBm (100 mW for Xbee). Receiver sensitivity typically ranges from -85 dBm to -100 dBm, depending on the specific module and operating conditions, providing connectivity up to several kilometers in line-of-sight.

When the speed and volume of data is constant, Bluetooth Low Energy (BLE) variants are widely used. This protocol is suitable for short distances, but signal amplifiers are also available that extend the range to several kilometers in the direct line of sight. BLE is a variant of Bluetooth designed for low-energy applications, such as smart bracelets, wearable devices, energy and water consumption sensors, sensors, security systems, miniature video recorders, etc. It offers very low power consumption. It is especially convenient for programmer interface extenders, as programmers do not have to sit in the

device operation area.

Last but not least, 4G/5G connectivity [11] significantly improves IoT, providing high speeds, low latency, and high scalability, which are essential for applications such as smart cities, autonomous vehicles, and industrial automation [12]. 5G speeds can reach up to 10 Gbps, with latency below 1 millisecond, and the network can support up to a million devices per square kilometer, which is practically unachievable by other systems. 5G uses spectrums such as Sub-6 GHz (typically between 1 GHz and 6 GHz.) and mmWave (typically in the range of 24 GHz to 100 GHz and higher), providing better coverage, and mmWave provides high speeds over short distances. NB-IoT (for small data devices) and Network Slicing (for virtualized networks that serve various IoT applications) technologies are key to efficient network operation. Among the main challenges are the costs of building a femtocell infrastructure, device compatibility, and the need for standardization to ensure seamless communication between different manufacturers [13]. Last but not least, the availability of miniature chipsets available for implementation in mass devices, as well as the upcoming implementation and support of 6G [14] will revolutionize IoT in the context of smart cities.

IoT “Big Data”

Studies show a linear increase in the number of IoT devices connected to the network globally, from around 13 billion in 2022 to over 42 billion in 2033 [15]. These devices will generate huge amounts of traffic, and the messages they send will need to be stored and processed. Investors are predicting over \$1.9 trillion for IoT by the end of this year [16]. The amount of data varies greatly depending on the type of device, its purpose, and the number of sessions. For example, simple temperature or humidity sensors may generate small amounts of data periodically, while devices such as video cameras or complex industrial machinery can produce large volumes of data in real time. According to some estimates, IoT devices are expected to generate over 79.4 zettabytes of data annually by 2025. This necessitates the need for efficient methods to collect, store, and analyze this data. To address this problem, cloud platforms such as AWS, Azure, and Google Cloud offer scalable storage and processing solutions.

Local and regional storage

Large cloud platforms such as Azure, Google Cloud, and AWS [17] offer scalable solutions for storing and processing IoT data, but they are not

always the most efficient choice. In many cases, local and regional storage provides better performance, lower costs, and higher security. This leads to a significant reduction in latency, especially important for real-time IoT devices (e.g. autonomous cars, industrial systems, medical applications) that require immediate data processing. Another advantage is the optimization of network load, IoT networks generate huge volumes of data. For example, industrial sensors, video surveillance, and smart cities can create terabytes of data daily, which must be analyzed, reduced, and filtered, with unnecessary records promptly destroyed. Last but not least, there is the issue of security, in industries such as healthcare, finance and government services, military there are strict regulations regarding data storage. The EU GDPR [18] requires that personal data be stored and processed within the EU.

GDPR is the EU's General Data Protection Regulation, which regulates the processing of personal data in the EU and imposes strict security and transparency requirements for data transfers outside the EU. IoT devices that collect personal data must comply with these requirements, ensuring the protection of users' rights. Metadata from devices can indirectly point to specific users, their habits and activities, which can lead to a lack of privacy.

APEC-CBPR (Asia-Pacific Economic Cooperation - Cross Border Privacy Rules) is a voluntary system for personal data protection used in the Asia-Pacific region. It allows data to be exchanged between countries, ensuring that local protection standards are respected. IoT devices in these regions can exchange data following these more flexible rules than those of the EU, which threatens the data privacy of users using cloud services outside the EU. When developing applications that process personal and metadata, manufacturers should also comply with the regulations of Australia - Privacy Act 1988, as well as Brazil and Latin America - LGDP, Africa largely copies EU legislation.

Independence from large suppliers (vendor lock-in) eliminates dependencies imposed over time by a specific platform. If migration to another system is necessary [19], the process can be difficult and expensive. Local and regional private repositories allow for a more flexible infrastructure that can be customized to the needs of the business. Processing acceleration can also be sought through embedded distributed solutions, such as:

Edge Computing allows data processing directly on devices or local hubs, rather than being sent to a central cloud. This is particularly useful for autonomous systems, smart factories, and low-latency devices [20].

Fog Computing extends this model by adding regional repositories that can process and store part of the data without burdening the cloud infrastructure. Relational databases with complementary organized textual (CSV – Comma Separated Values) and binary files organized on a file server (FTP – File transfer protocol), which is convenient at the stage of development of equipment, systems and scientific research, as well as working with video recordings, but leads to some limitations (see Table 4).

Table 4
Comparison of databases and file storage for IoT

Criterion	Text Files	Databases (InfluxDB, TimescaleDB)	Binary Files
Search Speed	Slow (linear search)	Fast (indexes, columns, segments)	Very fast (direct access and index)
Compression	Poor (large size)	High (10-100x smaller)	Additional, e.g., AVI container
Scalability	Issues with large data	Easily distributed across servers	Easy distribution, but file-dependent
Concurrent Access	Difficult (write conflicts)	Supports multiple sources	Difficult, no replication
API & Queries	Manual processing	SQL, REST API, aggregation functions	Specialized libraries for processing

Dedicated IoT databases

In general, cloud-connected IoT devices generate huge amounts of data, often in the form of time-series data, that must be stored, processed, and analyzed in real time [21]. Classic relational databases (SQL - ACID Atomicity Consistency Isolation Durability – Atomicity Consistency Isolation Durability) are not always optimal for these needs, which necessitates the use of specialized solutions [22] (see Table 5). In addition, with the increasing importance of distributed systems and blockchain technologies, it is necessary to take Considering these innovations for security, transparency and scalability in IoT infrastructure, the

main differences are given in Table 6. Key requirements for IoT databases with blockchain integration are related to:

- Time series processing – IoT data is time-ordered, which requires efficient storage, querying, and filtering mechanisms.
- High performance – the large amount of records requires optimized indexing and compression [23].

Table 5

Databases suitable for IoT

Database Type	Characteristics	Examples	Suitable for IoT?
Time-Series Databases	Designed for timestamped data, efficient queries, and compression	InfluxDB, TimescaleDB, OpenTSDB	Yes
NoSQL	Flexible structures, ideal for IoT metrics and JSON storage	MongoDB, CouchDB, DynamoDB, Cassandra	Possible
Key-Value Stores	Ultra-fast for storing sensor data and IoT caching	Redis, Apache Cassandra, Riak	Possible
Relational Databases	Suitable for IoT applications requiring a strict schema	PostgreSQL, MySQL, SQLite	Possible
Graph Databases	Ideal for IoT scenarios involving logistics, relationships, and dependencies between devices	Neo4j, ArangoDB	Ideal

- Scalability – IoT ecosystems are growing rapidly, which means that databases need to support horizontal and vertical scaling. Typically, a single hub for writing and replicated copies for reading.
- Lightweight architecture – Support for edge computing for data processing at the edge of the network.
- Reliability and security – IoT devices often operate in distributed environments and need resilience to data loss.
- Blockchain integration – To ensure data integrity, authenticity, and decentralization, which is especially important in sensitive and critical IoT applications such as healthcare [24] and logistics [25].

Examples of records

In the different types of databases used for IoT applications, records vary depending on the nature of the data and the storage and processing needs. Records in text files often contain a huge amount of repetitive information.

2025-02-09 10:01:00, Temp: 22.5°C, Hum: 45%
 2025-02-09 10:02:00, Temp: 22.6°C, Hum: 46%
 2025-02-09 10:03:00, Temp: 22.7°C, Hum: 47%

Each line contains a timestamp followed by the corresponding data, such as temperature and humidity. These labels may arrive at the same time as the data and be placed by the IoT module itself or added by the server software. While they have their uses and are easy to develop initially, they do not provide fast search or efficient capacity expansion for large volumes (see Table 6). A variation is the JSON format, which is also not particularly efficient in terms of volume and processing speed.

Table 6

Key features for SQL and Non SQL Databases for IoT

Characteristic	Standard SQL Databases (MySQL, PostgreSQL)	Time-Series Databases (InfluxDB, TimescaleDB, OpenTSDB)
Optimization for Time-Series Data	Not focused	Highly optimized (indexing, compression)
Write Speed	Medium speed	Extremely fast (LSM-Tree, Column Store)
Aggregation & Analysis	Manual programming	Built-in time-series analysis functions
Segmentation	Possible, but complex	Automatic (based on time)
Handling Large Data Volumes	Difficult to scale	Designed for scalability (distributed systems)
TTL (Automatic Data Deletion)	Manual	Automatically configurable

Time series records

Often, devices and applications require aggregation of long temporal series of measurement information, for example, medical ECG records, or data from industrial sensors, radargrams, etc. This data is conveniently converted from binary to BASE64 format, which allows it to be stored in regular text fields with up to 33% increased volume.

Time Series Records

Often, devices and applications require aggregation of long time series of measurement information, for example, medical ECG records or data from industrial sensors, radargrams, etc. This data is conveniently converted from binary to BASE64 format, allowing it to be stored in plain text fields with up to 33% increased volume. This is acceptable in many cases when it is not a question of terabyte volumes where file servers are used. Time series databases such as InfluxDB and TimescaleDB offer high performance, as well as aggregation and analysis features. They have built-in functions such as: avg() average, sum() sum, min() minimum value, max() maximum value, count() number of records, median() median, etc. They are widely used in IoT applications for real-time monitoring of sensors, networks, and devices. Example InfluxDB record:

```
measurement="temperature",      sensor_id="sensor01",  
location="room1" value=22.5 2025-02-09T10:01:00Z  
measurement="humidity",        sensor_id="sensor01",  
location="room1" value=45 2025-02-09T10:01:00Z
```

In this example, each record contains a measurement (e.g. temperature or humidity), tags (e.g. a unique sensor identifier, and location, which are clearly absent in text files because usually each sensor uses its own file, which makes managing them terribly complicated). A value (e.g. the measured temperature or humidity) and a timestamp (the moment of measurement in ISO format). In TimescaleDB, the records are similar, as it is a PostgreSQL extension for time series processing, with an example command:

```
INSERT INTO sensor_data (sensor_id, location,  
temperature, humidity, timestamp)  
VALUES ('sensor01', 'room1', 22.5, 45, '2025-02-09  
10:01:00');
```

InfluxDB supports time series compression using algorithms such as: TSD (Time Series Data) and Delta Compression. The system supports Retention Policies (RP) which allows defining retention policies that automatically remove old data that is no longer needed, reducing the load on the database. These policies can also include data compression for longer periods, which is not automatically achievable with sequential text files. The database also provides Downsampling - the ability to aggregate and compress longer periods of data by presampling, such as keeping only daily or weekly averages, instead of every value every minute.

TimescaleDB also supports data mobility, which provides built-in support for compression using hypertables (tables that expand automatically as new data is collected). Compressing old data allows for a significant reduction in storage requirements, while not compromising query performance. The database allows for automatic data aggregation via Continuous Aggregates, which allows for storing summarized values (e.g. average temperature per hour or day). Both systems have a variation of what's called Time Buckets, which are used to group data by time, such as hour, day, or week. These buckets allow us fast summarization for longer periods and reduce the load on the database when queries are repeated frequently.

Graph databases

A number of applications, such as: logistics, warehousing, transportation require databases storing a more specific type of data, which can be described by graphs containing nodes and edges. Such databases are, for example: Neo4j, ArangoDB, Amazon Neptune. The records in them can be used to monitor incoming requests from network infrastructures, suitable for sensor networks and mobile objects, etc.

Scalability and Distributed Storage

To handle large volumes of IoT data, both databases (InfluxDB and TimescaleDB) offer scalable solutions. This allows users to process and store data from thousands and millions of sensors in real time. The following technologies are used:

- **Sharding in InfluxDB** supports partitioning data into parts for distributed storage. This allows users to distribute data across different databases and increase scalability and performance.
- **Clustering:** is clustering support, which allows multiple servers to be combined to process large amounts of data and to ensure high availability.
- **Hypertables:** TimescaleDB uses the concept of hypertables, which are extended tables that are automatically partitioned by time intervals. This allows for scalable and efficient storage of large volumes of data and provides easy expansion.
- **Distributed Deployments:** TimescaleDB allows for distributed replication and data distribution across multiple servers, which provides high performance when processing large volumes of data.

Using Binary Files

Binary files are used to store streaming data in raw (RAW) and compressed formats, which provides high read and write performance [26]. They are typically used to store large amounts of data or data with variable structure, such as images or audio recordings from IoT devices that generate media data. Individual sections can be divided into chunks, which simplifies opening and positioning when reading and writing (AVI, MP3, WAV) with specialized software [27].

Conclusion

The review systematizes technologies and protocols suitable for cloud-based IoT systems with distributed large databases. Commonly used protocols in IoT are compared, as well as those based on wireless technology, such as Wi-Fi, Bluetooth, less often ZigBee and others, such as Ethernet and PLC. with their specifics, which will be useful for new designers of enterprise systems. Basic IoT protocols are grouped in terms of communication methods and where it is appropriate to do so, which will be useful for developers when choosing a specific solution: message-based, notification-based, packet-based. Special attention is paid to specialized IoT databases, different from relational databases offering a high degree of atomicity, consistency, isolation and immutability with built-in functionalities for reduction and automated aggregation and deletion of unnecessary records. These databases can be included in database courses at universities, thus diversifying the standard taught content of SQL-based applications with NoSQL optimized for time series, which will ensure the relevance of students' knowledge. Knowing the right technology solutions is crucial when choosing the equipment and software needed by designers of industrial and enterprise cloud-based IoT systems processing big data.

REFERENCES

- [1] Fernandes Nereus, 2020, Programmable Logic Controllers, PLC Overview, https://www.researchgate.net/publication/340428721_Programmable_Logic_Controllers_PLC_Overview
- [2] Enemosah, Aliyu & Ifeanyi, Ogbonna. (2025). SCADA in the Era of IoT: Automation, Cloud-driven security, and machine learning applications. *International Journal of Science and Research Archive*. 13. 3417-3435. 10.30574/ijrsra.2024.13.1.1975.
- [3] CLARE, W.N. & KAPLAN, G.T. & SADLON, D.R. & WIKTOROWICZ, A.C. & GILBERT, R.A. & Lipták, B.G. (1995). PLC—Programmable Logic Controllers. 10.1016/B978-0-7506-2255-4.50085-4.
- [4] Decotignie, Jean-Dominique. (2005). Ethernet-Based Real-Time and Industrial Communications. *Proceedings of the IEEE*. 93. 1102 - 1117. 10.1109/JPROC.2005.849721.
- [5] ESP32-C3 Wireless Adventure, A Comprehensive Guide to IoT, Espressif Systems, June 12, 2023 <https://www.espressif.com/sites/default/files/documentation/ESP32-C3%20Wireless%20Adventure.pdf>
- [6] Jake Hertz, Exploring ESP32: from History to Application, July 25, 2023, Electronics Engineer at Flux <https://www.flux.ai/p/blog/exploring-esp32-from-history-to-application>
- [7] Abu Radia, Moham Neo4j ed & Nimr, M. & Kamel, Mahmoud & Atlam, Ahmed. (2024). Implementation of a Real-Time Data Acquisition System. 10.13140/RG.2.2.29086.75847.
- [8] Pistek, Peter & Hudec, Martin. (2020). Using SMS for Communication with IoT Devices. *Mobile Networks and Applications*. 25. 1-8. 10.1007/s11036-020-01520-8.
- [9] Caiazza, Chiara & Luconi, Valerio & Vecchio, Alessio. (2025). Energy consumption of smartphones and IoT devices when using different versions of the HTTP protocol. 10.48550/arXiv.2502.19997.
- [10] Stefan Kutsarov, Savremenni radiomoduli, Sp. Inzhenering Review – broj 2/2020, 01.04.2020, ISSN 1311-0470 (Стефан Куцаров, Съвременни радиомодули, Сп. Инженеринг ревю - брой 2/2020, 01.04.2020 ISSN 1311-0470)
- [11] R. Pasarelski, T. Pasarelska, Izsledvane na mrezhovite funktsii i referentnata arhitektura na 5G mobilni sistemi, Sbornik statii ot nauchna konferentsia „ZNANIE,NAUKA,INOVATSII,TEHNOLOGII”, str. 388-400, ISSN 2815-3480, 6-7 yuli 2023 godina, Veliko Tarnovo, 2023 (Р. Пасарелски, Т. Пасарелска, Изследване на мрежовите функции и референтната архитектура на 5G мобилни системи, Сборник статии от научна конференция „ЗНАНИЕ, НАУКА, ИНОВАЦИИ, ТЕХНОЛОГИИ“, стр. 388-400, ISSN 2815-3480, 6-7 юли 2023 година, Велико Търново, 2023)
- [12] Adewale, Tunmise & Paul, Joel. (2024). AI, 5G, and IoT: How These Technologies Are Creating the Perfect Storm for Smart Systems. https://www.researchgate.net/publication/385855348_AI_5_G_and_IoT_How_These_Technologies_Are_Creating_the_Perfect_Storm_for_Smart_Systems
- [13] Stefan Kutsarov, "Bezzhichni komponenti za IoT prilozheniya", sp. Inzhenering Review, broj 1/2019, 05.03.2019, ISSN 1311-0470 (Стефан Куцаров, Безжични компоненти за IoT приложения, Сп. Инженеринг ревю - брой 1/2019, 05.03.2019, ISSN 1311-0470)
- [14] T. Pasarelska, P. Tzvetkov, R. Pasarelski, "Research approach and spectrum allocation analysis for 5G network development", 33rd International Scientific Symposium Metrology and Metrology Assurance 2023, p. 15-20, ISSN 2603-3194, 7-11 September, 2023

- [15] Vladkov E., The 6G future: a survey of technological promises and challenges. *Electrotechnica & Electronica (E+E)*, Vol. 59 (1), 2024, pp.9-16, ISSN: 0861-4717 (Print), 2603-5421 (Online)
- [16] Satyajit Sinha, State of IoT 2024: Number of connected IoT devices growing 13% to 18.8 billion globally, September 3, 2024, <https://iot-analytics.com/number-connected-iot-devices/>
- [17] Fabio Duarte6, 0+ Amazing IoT Statistics (2024-2030), September 19, 2024, <https://explodingtopics.com/blog/iot-stats>
- [18] Borra, Praveen. (2024). An overview of cloud data warehouses Amazon Redshift (AWS), Azure Synapse (Azure), and Google BigQuery (GCP). *International Journal of Advanced Research in Computer Science*. 15. 23-27. 10.26483/ijarcs.v15i3.7099.
- [19] Sullivan, Clare. (2019). EU GDPR or APEC CBPR? A comparative analysis of the approach of the EU and APEC to cross border data transfers and protection of personal data in the IoT era. *Computer Law & Security Review*. 35. 10.1016/j.clsr.2019.05.004.
- [20] Han, Jungsu & Park, Sun & Kim, Jongwon. (2020). Dynamic OverCloud: Realizing Microservices-Based IoT-Cloud Service Composition over Multiple Clouds. *Electronics*. 9. 969. 10.3390/electronics9060969.
- [21] Kuchuk, Heorhii & Malokhvii, Eduard. (2024). INTEGRATION OF IOT WITH CLOUD, FOG, AND EDGE COMPUTING: A REVIEW. *Advanced Information Systems*. 8. 65-78. 10.20998/2522-9052.2024.2.08.
- [22] Amghar, Souad & Cherdal, Safae & Mouline, Salma. (2018). Which NoSQL database for IoT Applications?. 131-137. 10.1109/MoWNet.2018.8428922.
- [23] Asiminidis, Christodoulos & Kokkonis, George & Kontogiannis, S.. (2018). Database Systems Performance Evaluation for IoT Applications. *International Journal of Database Management Systems*. 10. 01-14. 10.5121/ijdms.2018.10601.
- [24] Özmen, Ahmet & Ertugrul, M. (2024). Comparative Analysis of Database Technologies for IoT Environments.
- [25] Al-Nbhany, Wafaa & Zahary, Ammar & Al-Shargabi, Asma. (2024). Blockchain-IoT Healthcare Applications and Trends: A Review. *IEEE Access*. PP. 1-1. 10.1109/ACCESS.2023.3349187.
- [26] Vovchak, Orest & Veres, Zenovii. (2024). Blockchain Applicability for Storing Iot Telemetric Data in Logistic. *Advances in Cyber-Physical Systems*. 9. 164-169. 10.23939/acps2024.02.164.
- [27] Khan, Samiya & Alam, Mansaf. (2019). File Format s for Big Data Storage Systems. *International Journal of Engineering and Advanced Technology*. 9. 2906-2912. 10.35940/ijeat.A1196.109119.

Assoc. Prof. Dr. Georgi Petrov, telecommunication, system engineering, digital signal and image processing, industrial IoT, avionics and embedded systems design, BS, MS and PhD in telecommunication systems engineering at New Bulgarian University,
tel.:+35928110609 e-mail: gpetrov@nbu.bg

Msc. Eng. Velimir Vasilev, telecom and industrial engineering, independent consultant, BS in industrial management, MS in telecommunication systems engineering at New Bulgarian University and, PhD student in Technical University of Sofia, department Electrical measurement systems.
tel.:+359877178913 e-mail: velimir.vasilev@gmail.com

Received on: 26.02.2025