**Real-Time Monitoring**

Real-time monitoring is the delivery of continuously updated data about systems, processes, or events. Such monitoring provides information streaming at zero or low latency, so there is minimal delay between data collection and analysis.

(Barney, 2023)

Real-time monitoring is the process of collecting and storing performance metrics for data as it traverses your network. It involves polling and streaming data from infrastructure devices so that you know how your networks, applications, and services are performing. It’s the process of continuously gathering data that you can use to quickly jump into action when problems arise. But there wasn’t always a need for this.

**Importance -** This critical gap between the time to report and the time to take action can cost you a lot of money. Real-time monitoring helps fill this gap by giving you real-time data, alerts, and notifications so you can take the timely discourse to deal with any issue. -

**Types of real-time data:**

* Central processing unit (CPU) and memory utilization.
* Application response time.
* Service availability.
* Network latency.
* Network security.
* Web server requests.
* Transaction times.

**Example:** Fleet management - Fleet operations can use real-time data collection through vehicle fleet management software to catch reckless, sleepy, and impaired drivers, as well as maximize fleet use.

(Barney, 2023)

**Communication Protocols:**

A communications protocol is a set of formal rules describing how to transmit or exchange data, especially across a network. A standardised communications protocol is one that has been codified as a standard. Examples of these include WiFi, the Internet Protocol, and the Hypertext Transfer Protocol (HTTP).

You can use standardised communication protocols to make your data available, for example:

* from a web server via HTTP using a browser
* from a file server via File Transfer Protocol (FTP) using an FTP client application
* through a well-documented Application Programming Interface (API).

**Types** – File Transfer Protocol (FTP), TCP/IP, User Datagram Protocol (UDP), Hypertext Transfer Protocol (HTTP), Post Office Protocol (POP3), Internet Message Access Protocol (IMAP), Simple Mail Transfer Protocol (SMTP).

(Rouse, 2023)

HTTP, originally designed for web page transmission, has evolved into a versatile protocol for exchanging various forms of information, notably through web APIs (Application Programming Interfaces). These APIs enable computer applications to seamlessly share and access machine-readable data across the internet, with the ability to operate from remote locations, relying on complementary network protocols. Notably, APIs can facilitate the selective sharing and retrieval of specific data subsets, a crucial feature for managing unwieldy or sensitive datasets. Many established APIs, such as OGC WMS for map images and OAI-PMH for repository metadata, are well-documented standards for data and metadata exchange. In cases where a standardized API doesn't exist for a particular type of data or metadata, software developers can create custom APIs.

**API**

An API, or application programming interface, enables interaction between software applications, systems, or platforms to send and receive data. An API is a messenger that delivers your data request to an external source and then returns their reply to you. (airfocus, n.d.)

**Examples:** Social Media Login, PayPal Transactions, Price Comparison for Vacations.

**How It Works:**

APIs work by acting as the link between the user and the service provider, with the API software facilitating an exchange to take place. Whether that is an information request, an e-commerce transaction, sending money to a bank account, or any other situation requiring a two-way data transfer.

When you perform an action on a website, the API will send online data related to your action to its server. Subsequently, the server receives and interprets the data, completes the necessary activity, and then returns the outcome to the application. The application interprets the outcome data and shows you the answer in an understandable format.

(Airfocus, n.d.)

**What is an API and When Should it be Used?**

Although HTTP was initially developed to transmit web pages, it has since been adopted for transfer of other types of information too. One way to exchange information is through web APIs (Application Programming Interfaces). APIs allow computer applications to share and access machine-readable data. These applications can run on computers located anywhere, relying on other network protocols in the stack (see figure above) to handle data transport.

An API can also allow applications to share and access subsets of data. This is particularly useful for datasets where it would be impractical or unsuitable to transfer them in their entirety, such as large or sensitive datasets.

There are already well-documented APIs used for the exchange of data and metadata. For example, OGC WMS is used for geo-registered map images and OAI-PMH is used for exchanging repository metadata**.**

# (Wang et al., 2020)

The data capture architecture of this study is described in Figure 4. Data are captured from physical space. The real-time operation data have been accessed through a PLC and displayed on the HMI. PLC is connected to HMI using CN1 cable. V-SFT is used to create the interface to display machine data on HMI. HMI displays the status of the machine, such as engine speed, pressure from the cutting plate, operating time, output, and input and output platform height. HMI also connects with the digital dashboard for real-time monitoring. The digital dashboard is created with J-Mobile. J-Mobile is a software used to create a platform that completely covers connectivity, device management, process management, and data visualisation for the lower levels of industrial IoT platform architecture. JMobile can support this study, given that the machine under investigation is only equipped with a low level of iIoT capability. The digital dashboard can display the status, OEE, order scheduling, and alarm functions of the machine. The digital dashboard receives data from HMI, whereas HMI receives data from PLC. For machines with high IoT levels, several studies used AutomationML, OPC UA, and MTConnect for network configuration. This study uses Ethernet way for network configuration, given that the conventional machine such as DCM has limited IoT levels. HMI supports the Ethernet method to exchange data with the digital dashboard. The communication protocol used to connect the digital dashboard and HMI is Modbus TCP/IP, which uses an Ethernet protocol to exchange data to satisfy the real-time and reliability requirement. Modbus TCP/IP is the second category of industrial networks (Lu et al. 2020). The digital dashboard is connected to the database that stores order data. A mutual communication exists between the database and the machine through the digital dashboard. The data stored in the company’s database can be accessed and displayed on the digital dashboard to support the operations of a machine. Machine’s data on the dashboard can also be retrieved by the digital dashboard and then stored in DB1 in real time. The proposed DT design framework is a typically ideal pattern of DT. However, owing to the security consideration, the machine is operated by the customer side with confidential receipts that are not allowed to be changed externally. Thus, the data access/modification from the digital dashboard to the machine is disabled in practice. This implementation on DCM becomes a type of digital shadow.

# (Khan et al., 2020)

## Communication protocols for IIoT

In this section, we devise a taxonomy demonstrating the working of various communication protocols of IIoT.

Meng et al. [18] have proposed a ZMQ messaging design model which represents a generic and flexible Machine-to-Machine (M2M) messaging mechanism between the machines for event and command notification and data sharing. The experimentation using a case study of Quality inspection microwave sensor of food manufacturing production concludes that the proposed ZMQ technique is promising tool to deal with machine connectivity, machine presence and discovery, and messaging to allow ubiquitous data access and data interaction for rich sensing IoT application. The proposed technique solves the complex structure and heterogeneity problems of IIoT applications and contributes to cross-platform capability that allows the implementation on various powerful computers and light-weight devices.

Yang et al. [19] have first proposed two types of time synchronization attacks in IIoT called Absolute Slot Number (ASN) attack and Timeslot Template (TT) attack and then two algorithms called Sec\_ASN algorithm and Threshold Filter (TOF) have been proposed to counter the proposed two attacks using IEEE802.15.4e-based IIoT protocol stack. When new nodes join the network, they can receive incorrect values of ASN, under ASN attack. On the other hand in TT attack the malicious node misguide the legitimate node for calculating the error clock offset. The Sec\_ASN is the combination of authentication and a method called 2s + 1. The authentication is achieved through two steps, first verifying the information about the sender and then checking the sent information for tampering during communication. For the method 2s + 1, one node is selected from neighboring nodes as time parent node for synchronizations. TOF algorithm is proposed for clock offset estimation using least squares method through the difference between normal node times and sending time of the node.

Qiu et al. [20] have proposed a robust time synchronization scheme known as R-Sync which eliminates the isolated nodes to makes all nodes synchronized and also reduces energy consumption on entire synchronization process. Two timers are adopted to pull isolated nodes to join the synchronized networks. One timer is for time synchronization using two-way message exchanges and another timer at the beginning of the synchronization process. The authors have also introduced a root node selection algorithm to balance energy consumption among sensor nodes and extend the lifetime of sensor networks. The proposed algorithm is compared with three existing time synchronization algorithms, Timing-sync Protocol for Sensor Networks (TPSN), Groupwise Pair selection Algorithm (GPA), and Spanning Tree-based Energy-efficient Time Synchronization (STETS) and through experimentation it is shown that the proposed R-Sync algorithms has lower energy consumption than GPA, TPSN and STETS algorithms, especially in densely connected and large scale networks.

Katsikeas et al. [21] studied the security implementation of MQTT (Message Queue Telemetry Transport) protocol using payload encryption (with AES, AES-CBC, AES-OCB) and link layer (with encryption with AES-CCM) in industrial domain. The authors evaluated and compared the secure and lightweight MQTT implementation using WSN testbed (Raspberry Pi) and through simulator. Two nodes are used during evaluation process, Publisher to emulate IIoT sensors and encrypt the data, and Subscriber to emulate IIoT actuators and decrypt the data. For comparison, latency, memory usage and energy consumption are considered. It is observed that MQTT implantation payload encryption (with AES, AES-CBC, AES-OCB) required more memory, energy and high latency as compare to MQTT implantation with link layer (with encryption with AES-CCM). However, if payload size is limiting factor, AES-CBC could be a better option.

Ferrari et al. [22] have investigated the latency of MQTT protocol for IIoT by observing the round trip time (RTT) through transferring data from the field to the Clouds and back. The authors have used embedded device IoT2040 from Siemens, energy saving Intel Quark x1020 (+secure boot), 1 GB RAM, 2 ethernet ports, 2xRS232/485 interfaces, battery backed RTC, Yocto Linux and industrial PC Intel i3-5000 with Windows 7 for the experimentation. The experimental works conclude that intercontinental roundtrip latency is less than 300ms, while local roundtrip latency is achieved at less than 50ms. The roundtrip delay is caused by the free Clouds used, internet connection, and the used hardware. However, implementation of filter reduces the values effectively.

Kiran et al. [23] have proposed a novel Markov chain based analytical/theoretical model to analyze the performance of unslotted Prioritized Contention Access (PCA) and Carrier-sense Multiple Access with Collision Avoidance (CSMA/CA) in nonbeacon-enabled PAN and slotted PCA and CSMA/CA in beacon enabled Personal Area Network (PAN). The reliability and the performance of the proposed model with less than 5% error is validated using Monte Carlo simulation and real-time test bed. The achieved results of slotted PCA claim that the reduction of 63.3% and 97% in delay and power consumption respectively compared with the slotted CSMA/CA, whereas unslotted PCA achieves reduction of 53.3% and 96% for delay and power consumption, respectively compared with unslotted CSMA/CA without significant loss of reliability.

# (Zhou et al., 2022)

In system-level data interaction, developers usually encounter the problem of low data collection openness by designing different drivers to collect status data of devices from different hardware suppliers. For the monitoring system of the RMGC, most of the data are captured in real time and the source is complex. In addition, data interaction between the digital twin RMGC and physical equipment is required, and then the interaction and integration between virtual and real data becomes more difficult. OPC UA has a low average communication delay. Ferrari et al. [33] verified that the average cost of data transmission from a Siemens S7 1500 Controller to an IBM Bluemix Platform is 70 ms. Lee et al. [34] used OPC UA to establish a real-time data transmission and reception connection between two-wheeled robots, with an average delay time of 20 ms. In addition, Mühlbauer et al. [35] verified that OPC UA has excellent scalability, which is determined by the information model of OPC UA and its client–server response mechanism. In this paper, the open OPC UA protocol is used for data interaction [36,37,38] between the monitoring host and the controller to break through the communication barriers between the virtual and the real, or between different hardware devices. The data interaction in the whole monitoring process.

The OPC UA server script embedded in the IPC reads the data required by the monitoring system in real time, and conducts preprocessing steps on the data after acquiring it, such as filtering, unifying the type, and structuring after removing the unit, and then transmits it to the software platform via ethernet. After the OPC UA client script in the monitoring computer obtains the data uploaded by the server and decodes the data, the computer CPU starts multi-threading, and sends the data to the digital twin RMGC through the virtual and real interactive data script for synchronous mapping simulation, and to the SQLite database through the data storage script for data storage. The OPC UA client script can send the operator’s control instructions to the server, and then convert them into control codes through the control program to control the equipment. In addition, to facilitate offline reproduction of historical working conditions, the digital twin RMGC should have the ability to extract historical data from the SQLite database for offline simulation.

# (Bayılmış et al., 2022)

**Popular protocols in IoT application layer**

3.1. REST

REST is a term used to describe an architectural style for network systems in R.T. Fielding's Ph.D. It is an operating system and language independent architecture for the design of network applications that uses simple HTTP to connect devices. In other words, REST is not a standard, it is an architectural approach that includes standards such as HTTP, Uniform Resource Locator (URL), EXtensible Markup Language (XML), etc. It has Peer-to-Peer (P2P) communication structure based on stateless client/server and request/response model. The services that use REST architecture are generally called RESTful services. It is widely used in web-based applications because it is simple, flexible, and easily extensible. All operations with other web services can be done with RESTful services. RESTful accesses the cloud over Transmission Control Protocol/Internet Protocol (TCP/IP) and it provides Hypertext Transfer Protocol Secure (HTTPS). In message format, the header is in American Standard Code for Information Interchange (ASCII) and the payload is in XML, JavaScript Object Notation (JSON), and HTML structures [[45](https://www.sciencedirect.com/science/article/pii/S2352864822000347#bib45)].

The REST architecture is a web-oriented architecture that enables data transfer in the client-server model over the HTTP protocol. The data format is flexible due to the fact that it can be modified according to the desired additional features and purposes. In the REST architecture, each component is defined as a resource. These resources can be a text file, HTML page, image, video, or any type of data. The REST server makes each resource accessible by assigning a Uniform Resource Identifier (URI). Clients can reformat or use them directly by accessing them through URLs created with URIs. There is no restriction on the resource format in REST architecture. The data can be transmitted in many different formats depending on the application. These formats can be plain text, JSON, or XML, etc., commonly used in Internet [[46](https://www.sciencedirect.com/science/article/pii/S2352864822000347#bib46),[47](https://www.sciencedirect.com/science/article/pii/S2352864822000347#bib47)].

In the REST architecture, messages are transmitted in the request-response system via HTTP. The client sends the message as an HTTP request and the server responds with an HTTP response. This model is illustrated in [Fig. 4](https://www.sciencedirect.com/science/article/pii/S2352864822000347#fig4). The REST architecture recommends the use of Get, Post, Put, and Delete methods for HTTP requests. These methods may become important in terms of application stability and safety. REST is one of the most frequently used architecture in IoT with its easy access to data, flexible structure and small frame size [[46](https://www.sciencedirect.com/science/article/pii/S2352864822000347#bib46)].



Fig. 4. Message transmission model in REST architecture.

3.2. MQTT

MQTT is a messaging transport protocol that is based on publish-subscribe architecture. The first version of the MQTT protocol was introduced in 1999 by Andy Stanford-Clark and Arlen Nipper [[48](https://www.sciencedirect.com/science/article/pii/S2352864822000347#bib48)]. The version MQTT v3.1 was adopted by OASIS consortium in 2013 and also certified by International Organization for Standardization (ISO) and the International Electrotechnical Commission (ISO/IEC) JTC1 in 2016. The current version v.5.0 and MQTT for Sensor Network (MQTT-SN) are the variation of the protocol to support [wireless sensor networks](https://www.sciencedirect.com/topics/engineering/wireless-sensor-network). It is developed for resource constrained devices that aim for low cost, open source, reliability, simplicity [[49](https://www.sciencedirect.com/science/article/pii/S2352864822000347#bib49)].

MQTT consists of clients that communicate via brokers and distributes the messages between clients. MQTT client may take the role of a publisher or a subscriber. The message information includes data and topic. Clients communicate with each other through the corresponding topic that is distributed by the broker. Thus, the MQTT broker stores and publishes the information from the publisher to the subscribers. [Fig. 5](https://www.sciencedirect.com/science/article/pii/S2352864822000347#fig5) shows the MQTT communication model.

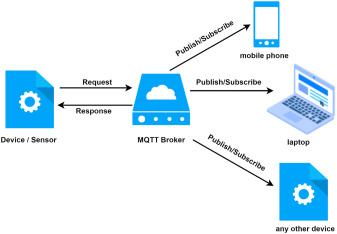


Fig. 5. MQTT messaging structure.

There are three levels of QoS in MQTT protocol, including at most once, at least once, exactly once for different application requirements. MQTT uses TCP/IP for data transmission and Transport Layer Security/Secure Sockets Layer (TLS/SSL) to provide communications security. The message size can vary according to application requirements up to 256 ​MB with a 2-bytes header. It has a wide range of current platforms such as [Amazon Web Services](https://www.sciencedirect.com/topics/engineering/amazon-web-services), Microsoft Azure IoT, Adafruit, Facebook Messenger, etc.

# References

Airfocus (no date) *What is an API? API definition, examples, benefits, Challenges & FAQ*, *What Is an API? API Definition, Examples, Benefits, Challenges & FAQ*. Available at: https://airfocus.com/glossary/what-is-an-api/#:~:text=APIs%20empower%20developers%20to%20be,interface%20with%20other%20developers’%20applications (Accessed: 04 October 2023).

Barney, N. (2023) *What is real-time monitoring?: Definition from TechTarget*, *WhatIs.com*. Available at: https://www.techtarget.com/whatis/definition/real-time-monitoring#:~:text=Real%2Dtime%20monitoring%20is%20the,performance%20issues%20and%20critical%20events (Accessed: 29 September 2023).

Bayılmış, C. *et al.* (2022) ‘A survey on communication protocols and performance evaluations for internet of things’, *Digital Communications and Networks*, 8(6), pp. 1094–1104. doi:10.1016/j.dcan.2022.03.013.

Khan, W.Z. *et al.* (2020) ‘Industrial internet of things: Recent advances, enabling technologies and open challenges’, *Computers &amp; Electrical Engineering*, 81, p. 106522. doi:10.1016/j.compeleceng.2019.106522.

Rouse, M. (2023) *What is a communication protocol? - definition from Techopedia*, *Communication Protocol*. Available at: https://www.techopedia.com/definition/25705/communication-protocol (Accessed: 04 October 2023).

Zhou, Y. *et al.* (2022) ‘A digital twin-based Operation Status Monitoring System for port cranes’, *Sensors*, 22(9), p. 3216. doi:10.3390/s22093216.