UDC 004.588:681.5
DOI: 10.15587/1729-4061.2026.363630

EVALUATION OF EDGE-BASED DATA STREAM PROCESSING SCENARIOS IN INDUSTRIAL AUTOMATION SYSTEMS: DATA REDUCTION, LATENCY, AND RESOURCE UTILIZATION

Igor Krasnikov
Corresponding author
Candidate of Technical Sciences, Associate Professor*
E-mail: ihor.krasnikov@khpi.edu.ua
ORCID: https://orcid.org/0000-0002-7663-1816
Kyrylo Halliamov
PhD Student*
ORCID: https://orcid.org/0009-0007-8762-4885
Ihor Lysachenko
Candidate of Technical Sciences, Associate Professor*
ORCID: https://orcid.org/0000-0002-3723-8587
*Department of Automation of Technological Systems and Environmental Monitoring
National Technical University «Kharkiv Polytechnic Institute»
Kyrpychova str., 2, Kharkiv, Ukraine, 61002

This study explores the stream of process measurements after processing at the edge node in the chain "field-level equipment → edge node → SCADA → database". The task addressed relates to the insufficient quantitative assessment of the impact of processing scenarios at the edge level on the stream intensity, time characteristics, and load of system nodes.

This paper considers an edge-based approach to the stream processing of process data from industrial automation systems. 5 scenarios for processing the stream of process data at the edge node have been considered. A comparative analysis was performed by the number of records after processing, end-to-end latency, reduction ratio, jitter, and resource load. To that end, a bench was implemented that reproduces data transmission from field-level equipment through an edge node to the SCADA system and database. The basic mode without optimization, filtering by the change threshold, periodic selection, aggregation in a time window, event filtering, and a hybrid scheme were investigated.

All edge processing scenarios reduced the stream intensity compared to the baseline. The largest reduction was provided by EV and DB: 0.976 and 0.962. For most scenarios, the end-to-end latency was 71.064–81.197 ms, for HYB – 112.457 ms. The lowest jitter was obtained for EV – 25.98 ms and DB – 38.521 ms. This is explained by the fact that event and threshold selection cut off background signal changes. Preprocessing reduces the network load without a noticeable increase in the load of the edge node. This makes it possible to consider processing at the edge node as a means of data reduction and formation of a controlled stream for the SCADA system and database.

The practical significance implies using the results for comparative selection of processing scenarios for process data taking into account network traffic, latency stability, as well as edge node resources

Keywords: process data, stream processing, edge processing, SCADA system, stream reduction

Received 16.03.2026
Received in revised form 22.05.2026
Accepted date 01.06.2026
Published date 30.06.2026

How to Cite: *Krasnikov, I., Halliamov, K., Lysachenko, I. (2026). Evaluation of edge-based data stream processing scenarios in industrial automation systems: data reduction, latency, and resource utilization. Eastern-European Journal of Enterprise Technologies, 3 (2 (141)), 101–110. https://doi.org/10.15587/1729-4061.2026.363630*

1. Introduction

Modern industrial systems generate more and more process data. This relates to the evolution of Industry 4.0 and smart factory approaches when equipment, sensors, control systems, and information services must work more coherently [1, 2]. Data come from sensors, controllers, and other automation elements. They are used not only for storage or analysis after the process is completed. In many cases, they need to be processed immediately to quickly notice deviations, monitor the condition of the equipment, and support the operation of the operator or control system.

This is why interest in streaming data processing is growing. This approach makes it possible to work with measurements as they arrive, rather than waiting for them to be stored in a database. For industrial tasks, this is useful because some operations, such as filtering, aggregation, or event search, can be performed almost in real time.

Edge computing plays a special role here. It enables transferring some of the processing closer to the place where the data are acquired. This can reduce latencies, network load, and the amount of data that needs to be transmitted further into the system.

At the same time, using stream processing in industrial systems is not so simple. Process data are associated with real physical processes, so temporal consistency, stability of operation, and predictable latency are important for them. Because of this, it is necessary to understand which operations are advisable to perform at the edge level, how they affect the volume of data, and what resources are needed.

Thus, the large volume of process data, the need for their prompt processing and reducing the load on the network and

information infrastructure predetermine the relevance of research aimed at edge stream processing.

2. Literature review and problem statement

Data stream processing is one of the important directions for developing industrial information systems that work with continuous flows of process measurements and events under a mode close to real time. In industrial automation systems, this is of particular importance because process parameters are formed continuously, have a time reference, and are used for monitoring, diagnostics, archiving, and decision support.

Work [3] reports the results of generalizing the development of data stream processing systems. It is shown that modern data stream processing systems have formed as a separate class of software systems, in which the order of arrival of events, work with time stamps, window processing, state management, scalability and correctness of results under the condition of continuous data stream are important. At the same time, the applied use of these approaches for processing process data streams in industrial automation systems has not been considered in detail. This limitation is due to the fact that the general theory of stream processing considered mainly universal event streams, while the connection of process measurements with physical processes, the SCADA layer, and the long-term data storage subsystem remained overlooked.

In [4], the results of evaluating the performance of real-time streaming data processing systems for IoT applications are given. It is shown that response time, throughput, jitter, and scalability are important for such systems. The authors proposed a four-layer IoT infrastructure with a streaming layer and compared five distributed systems in terms of performance indicators. However, the work did not separately assess the impact of specific classes of preprocessing operators on data transmission in the industrial chain. This was explained by the broader nature of IoT applications, while for industrial automation systems the process significance of signal changes, latency stability, and coordination with the SCADA layer and database were important.

The results from [3, 4] showed that general approaches to data streaming have already been worked out in detail.

An option to overcome their limitations for industrial automation may be to move from the analysis of general streaming platforms to the evaluation of specific processing scenarios at the edge level of the process data stream. However, the cited papers did not consider an application model for the chain "field-level equipment → edge node → SCADA → database" and did not assess the impact of various operators on the reduction of the processed stream, end-to-end latency, jitter, and resource load.

Work [5] reports the results of a study on process-oriented and streaming data processing of industrial sensors. It is shown that industrial sensor data should be processed in the form of a sequential chain of operations, which includes the capture, processing, storage, and visualization of primary data. The proposed Sensor Processing Pipeline approach is important as it takes into account the differences in industrial data format, sampling rate, timestamps, and process-relatedness. However, the paper does not address in detail the impact of different pre-processing operators on throughput, timing, and system node load. This is because the focus is on the organization of the sensor data pipeline, while a comparative evaluation of several edge processing scenarios in a multi-level industrial architecture is not performed.

Paper [6] reports the results from the analysis of multidimensional data on the Industrial Internet of Things in Edge-Fog-Cloud architectures. It is shown that in modern IIoT systems, data has not only a large volume but also a complex multidimensional structure as it contains various logical attributes, indicators, and characteristics of industrial processes. It also systematizes approaches to the analysis of such data by processing types and levels of calculation execution: edge, fog, and cloud. At the same time, the paper does not fully consider the choice of the level of execution of streaming operators and the quantitative assessment of their impact on the operation of the industrial system. This was due to the survey-architectural nature of the study, in which most attention was paid to the classification of approaches and levels of processing, rather than to the experimental comparison of operators in a specific industrial tract.

A summary of [5, 6] showed that industrial data requires not only storage but also pre-processing, taking into account the time of formation, production context, and the level of architecture at which this processing is performed. As an option to overcome the corresponding difficulties, the construction of an applied model is proposed, in which the process data stream is considered within the sequential chain "field-level equipment → edge node → SCADA → database". However, in the above papers, such a model was not separately formed, and the influence of different classes of operators on stream reduction, latency, jitter, and resource load was not studied comprehensively.

In [7], the results of analyzing the architectural principles of edge computing in the Industrial Internet of Things are reported. It is shown that transferring part of the processing closer to the data sources made it possible to reduce latencies, reduce the use of network bandwidth, and increase the responsiveness in the IIoT environment. However, the work did not determine which operators should be performed at the edge level for processing data and how to assess the consequences of such a transfer for the SCADA layer and database. This was due to the fact that architectural reviews of edge computing mostly focused on the general principles of computing distribution, while the evaluation of operators in a specific industrial chain remained outside the scope of consideration.

In [8], the results of a practical study on edge processing of IoT streams using Raspberry Pi and Node-RED-based infrastructure are given. It is shown that stream preprocessing can be implemented even on limited computing resources, which is important for building affordable edge solutions. At the same time, the work did not separately consider the integration of the edge node with the SCADA level and the relational database, as well as the time and resource consequences of various processing scenarios. This was explained mainly by the applied nature of the study, which did not set the task of comparing typical classes of stream operators in a complete industrial tract.

Papers [7, 8] confirmed the feasibility of transferring part of the processing to the edge level. An option for overcoming the limitations of those studies is the experimental verification of several typical scenarios of edge processing under the same conditions and within a single chain of transmission and storage of process data. However, the authors do not provide a comparative analysis of such scenarios according to the criteria of reduction of the processed stream, end-to-end latency, jitter, and resource profile of the system nodes.

For multi-tier industrial architectures, the choice of data transfer protocols is also important. In [9], the results of a study on the suitability of the communication protocols MQTT, AMQP, Kafka, ZeroMQ and OPC UA for transmitting high-frequency industrial streams in edge/fog/cloud environments are reported. It is shown that latency and jitter are key characteristics that directly affect the temporal behavior of the system. However, the work did not separately study the influence of the type of edge preprocessing on the temporal characteristics of data transfer. Most attention was on the protocol level, while the logic of forming the processed stream at the edge node remained outside the scope of the analysis.

In [10], the results of designing a distributed real-time data management system using edge computing, OPC UA, and Kafka are given. It is shown that these technologies could be used as part of a holistic architecture for data collection, transmission, and storage. However, issues related to comparing different scenarios of pre-processing of process data before transmission to the SCADA level and recording into the database remained unresolved. The reason was that the work focused on building an architecture for data exchange and management, rather than analyzing the impact of individual operator classes on stream intensity, latency, jitter, and resource load.

Papers [9, 10] have shown that for industrial systems, not only computational layers are important but also protocols, transmission timing characteristics, and data exchange architecture. As a way to overcome the limitations, it was proposed to combine the analysis of edge processing with the evaluation of the timing characteristics of the path "edge node → SCADA → database". However, those papers did not investigate how different preprocessing operators change the processed stream and how this affects the time and resource indicators of a multi-level industrial system.

Our review of the literature [3–10] demonstrates that current research quite fully reveals certain aspects of data streaming, industrial sensor streams, edge computing, Edge-Fog-Cloud architectures, and data transmission protocols. Papers on the theory of streaming systems consider the order of event processing in time, time stamps, windowing, preservation of intermediate state, scalability, and fault tolerance. In studies on industrial data and IIoT architectures, the feasibility of preprocessing data closer to the sources of their formation has been shown. In studies on the protocol level, the importance of latency and jitter for multi-level industrial systems has been confirmed.

It should be noted separately that the reduction of measurement data redundancy can also be considered within the framework of digital signal processing, in particular with the use of spectral analysis, DFT, DCT, and information entropy estimation. Such approaches are appropriate for the tasks of compression, signal reconstruction, and assessment of its information completeness. Our work considers another statement of the problem, related to the preliminary formation of the process data stream before its transmission to the SCADA system and database. The object of evaluation is the processed data stream after the use of edge operators.

At the same time, for industrial automation tasks, the reported results are not enough to make a reasonable choice of the edge stream processing scenario. Process data often contain noise, arrive with different intensities, have a time frame, and must preserve characteristic signal changes. In addition, the preprocessing of such data must be coordinated with the SCADA level and the data storage subsystem. Existing studies mostly focus either on general stream processing mechanisms, or the architectural principles of edge/fog/cloud computing, or the protocol level of data transmission. Given this, they do not consider in detail the comparison of typical edge operators in the chain "field-level equipment → edge node → SCADA → database" taking into account the complex time and resource characteristics of the processed stream.

The industrial chain "field-level equipment → edge node → SCADA → database" combines the physical nature of process signals, the limitations of real or near real time, the requirements for preserving technologically significant changes, and the resource limitations of edge nodes. In addition, the passage of data through the SCADA level and the storage subsystem generates additional time and resource effects that cannot be fully assessed only by isolated analysis of a single operator. Therefore, the direct transfer of general stream approaches without experimental comparison of operators does not provide reasonable grounds for choosing a practical processing scenario.

Therefore, the task to quantitatively assess the impact of various edge processing scenarios on the processed data stream, the time characteristics of the transmission path, and the resource profile of the nodes of the industrial automation system remains unresolved. This is precisely what necessitates the comparative analysis of typical scenarios for edge stream processing in the chain "field-level equipment → edge node → SCADA → database" according to the criteria of reduction of the processed stream, end-to-end latency, jitter, and resource load.

3. The aim and objectives of the study

The purpose of this work is to determine the impact of scenarios for edge stream processing of process data in industrial automation systems on the reduction of the processed stream, time characteristics, and resource load of nodes. This allows us to compare the suitability of the studied classes of stream operators within the adopted experimental configuration. To achieve this goal, the following tasks were set:

– to propose a generalized scheme for the passage and processing of process data in a multi-level industrial system;

– to implement an experimental bench and form a basic scenario and five scenarios for edge processing, representing different classes of stream operators;

– to evaluate the reduction of the processed stream and time characteristics when using different scenarios;

– to evaluate the resource profile of system nodes.

4. The study materials and methods

The object of our study is the process data stream after edge processing in the multi-level chain "field-level equipment → edge node → SCADA → database". The principal hypothesis assumes that transferring part of the stream processing operators to the edge level makes it possible to reduce the intensity of the processed stream and the network load without a noticeable increase in the computational load on the edge node. The study assumes that the transport path, SCADA configuration, and the mechanism for writing to the database remain the same for all scenarios, and the differences in the results are determined by the type of edge operator.

A basic simplification is the use of a laboratory bench with one process data source and specified parameters of the processing scenarios.

The methodological approach is based on the representation of process data as a time-ordered stream of measurements, to which reduction, discretization, aggregation, and event selection operators can be applied up to the stage of long-term storage. Within the framework of the study, the object of comparison between scenarios is the processed stream, that is, the stream after edge processing.

The experiment was conducted on a multi-node bench that reproduces a typical architecture of an industrial IIoT/SCADA chain. The field level is represented by a DHT11 sensor and an ESP32-S3 microcontroller, which performed temperature and relative humidity reading, timestamp generation, and message transmission via MQTT. The edge level was implemented on a Raspberry Pi 4 with Node-RED, where stream processing and publishing of the processed stream to MQTT topics were performed. The SCADA level was implemented based on Ignition, which received the processed stream and wrote it to MySQL via JDBC. A local NTP server was used to synchronize time between nodes. Resource telemetry of edge, SCADA, and DATABASE nodes was collected using Prometheus.

One baseline scenario and five edge optimization scenarios were investigated. In all cases, the transport path, SCADA configuration, and database write mechanism remained unchanged; only the streaming operator on the edge node changed.

The scenario parameters ($dT = 0.2$, $T_{out} = 5$ s, $W = 30$ s, etc.) were chosen as engineering representative test values for the comparative experiment. Their choice was aimed at ensuring a noticeable effect of reduction of the processed stream and at the same time preserving technologically significant signal changes within the limits of the test bench. The threshold parameters $dT = 0.2$ for temperature and $dH = 1$ for relative humidity were used as test limits of significant change for the slow process reproduced on the laboratory bench. Other parameters set the time intervals of selection, aggregation or periodic state update and were used to obtain comparable operating modes of different classes of operators.

The choice of these parameters was not based on the analysis of the spectral band of the signal because the frequency analysis and reconstruction of the original signal after reduction were not included in the scope of this work. Parametric, entropic, and spectral justification of scenario settings is considered as a separate direction for further research.

The base scenario A0 corresponded to the no-optimization mode when the stream was transmitted without reduction. The DB scenario implemented deadband filtering with parameters $dT = 0.2$ and $dH = 1$. The DS scenario implemented periodic sampling with parameter $T_{out} = 5$ s. The TW scenario implemented aggregation in a time window with parameters $W = 30$ s and *mean*. The EV scenario implemented event selection with thresholds $T_{high} = 28$ and $H_{high} = 70$. The HYB scenario combined deadband filtering and periodic heartbeat transmission with parameters $dT = 0.2$, $dH = 1$, $hb = 30$ s. This set of scenarios covers the main classes of operators used for reduction and structuring of the process data stream before its storage.

Each scenario was evaluated based on three valid measurements ($n = 3$). One measurement consisted of two phases: a warm-up lasting 60 s and a measurement lasting 600 s. The warm-up phase data was not used in the final calculations. The runs table in the database stored metadata for each measurement: run_id, scenario, operator, parameters, start time, warm-up duration, and measurement interval duration.

To uniquely match the records of the processed stream with a specific measurement, each message contained the run_id attribute, which was stored in payload_json. A measurement was considered valid if there was a non-zero number of processed records associated with the corresponding run_id within the measurement interval. Invalid measurements were replaced by additional runs until three valid repetitions were achieved for each scenario. All final calculations were performed on a fixed list of valid run_ids.

To assess the time characteristics of the path, time stamps were used that corresponded to the moments of message arrival at the edge node, exit from it after processing, arrival at SCADA, and insertion into the database. Based on them, the duration of edge processing, the latency edge node → SCADA, the latency SCADA → database, and the integral end-to-end latency from the completion of edge processing to the recording in MySQL were calculated. As a generalized measure of jitter, the standard deviation of the end-to-end latency values within the measurement was used.

The basic functional metric was the reduction ratio of the processed stream relative to the baseline scenario. For each measurement, the number of records of the processed stream within the measurement interval and the average stream intensity in records per second were determined. Then, at the scenario level, the average values for three valid measurements were calculated, and the reduction ratio was determined relative to the average number of records in the A0 scenario. This approach allows us to quantify the effect of an edge operator in reducing the amount of transmitted and stored data.

Resource metrics were obtained from Prometheus for edge, SCADA, and DATABASE nodes. CPU utilization, RAM usage, and network traffic for both receive and transmit were used for analysis. For the database server, only CPU and RAM metrics were available in the available dataset. All metrics were aggregated within the same 600-second measurement interval as the stream and time metrics and then averaged at the scenario level.

All summary metrics were reported in the format $\mu \pm \sigma$, where μ is the mean and σ is the standard deviation over three valid measurements. This format was used for the processed stream, time, and resource metrics. The focus was on the comparative analysis of scenarios by the degree of stream reduction, latency, jitter, and resource profile.

5. Results of investigating edge stream processing of process data

5. 1. Generalized scheme of data stream and processing in a multi-level industrial system

Fig. 1 shows a generalized data stream path in a multi-level industrial system: from the field level through the edge/PLC level and the SCADA level to the storage and integration level. At the field level, the formation of a stream of primary measurements was provided, and at the edge level – the possibility of applying filtering, selection, local threshold rules, or preliminary aggregation. After that, the data was transferred to the SCADA level and further to the storage subsystem.

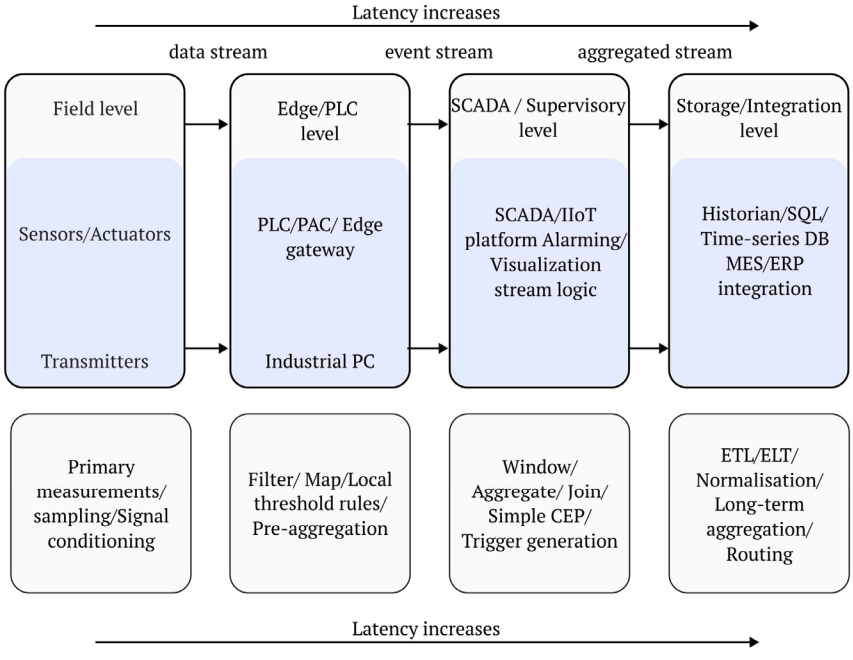


Fig. 1. Generalized scheme of data stream and processing in a multi-level industrial system

In our work, most attention is on the stream formed after the application of the edge operator. It was this processed stream that was used for further comparison of the scenarios. The differences between the processed and primary flows concerned the number of records, regularity of receipt and time characteristics. This provided the possibility of analyzing not only the fact of data delivery to SCADA and the database but also how the selected edge processing method changed the stream intensity, latency, and load on the system nodes.

The edge node in this model was considered as a level of preliminary transformation of process data. For quantitative analysis, the number of records of the processed stream per measurement interval, the average stream intensity, the reduction ratio relative to the baseline scenario, the time metrics of the path " edge node → SCADA → database", as well as the resource indicators of edge-, SCADA- and DATABASE-nodes were used.

5. 2. Implementation of the experimental bench and edge processing scenarios

One baseline scenario without optimization and five edge stream processing scenarios were implemented, given in Table 1. They differed in the type of operator applied to the data stream at the edge level.

The baseline scenario A0 was used as the initial mode for comparison. It corresponded to direct streaming without reduction. The DB, DS, TW, EV, and HYB scenarios covered different data pre-transformation techniques: threshold filtering, periodic sampling, time-window aggregation, event filtering, and a combined mode with heartbeat transmission.

Table 1

List of processing scenarios

Scenario	Operator/Parameter	Explanation
A0	NONE	Basic scenario without optimization; transit stream to SCADA and DB
DB	deadband $dT = 0.2$, $dH = 1$	Reduction of stream due to a significant change in parameter
DS	$T_{out} = 5$s	Periodic selection and stabilization of the stream frequency
TW	$W = 30$ s, mean	Aggregation in a time window with averaging
EV	$T_{high} = 28$, $H_{high} = 70$	Event filtering and message generation based on rules only
HYB	deadband + heartbeat 30 s	Combination of reduction and guaranteed periodic status updates

This set of scenarios provided the opportunity to compare the basic modes of edge processing of process data: direct transmission, threshold reduction, periodic delivery, temporal aggregation, event selection, and a combined mode between reduction and regular state update.

5. 3. Results of evaluating a reduction of the processed stream and time characteristics

A comparison of the scenarios showed that the use of edge operators in all the studied cases led to a decrease in the intensity of the processed stream compared to the baseline scenario A0. For the baseline scenario, the average number of stream records per 600 s was 597.3 ± 0.94, which corresponded to an average intensity of 0.996 ± 0.0016 records/s. For all optimization scenarios, these values were lower. The greatest reduction was provided by EV and DB, for which the reduction coefficient RR_{proc} was 0.9760 and 0.9615, respectively. For TW, this indicator was 0.9353, for HYB – 0.9079, and for DS – 0.8159. A comparison of the scenarios by the reduction coefficient is shown in Fig. 2.

In terms of the number of records of the processed stream, the lowest value was observed in the EV scenario – 14.33 ± 13.02 records per 600 s. For DB, this value was 23.00 ± 11.43, for TW – 38.67 ± 0.94, for HYB – 55.00 ± 1.41, and for DS – 110.00 ± 64.15. The generalized results of the stream reduction assessment are given in Table 2. Thus, event and threshold selection provided the greatest reduction in the intensity of the processed stream, while periodic selection formed a less reduced stream.

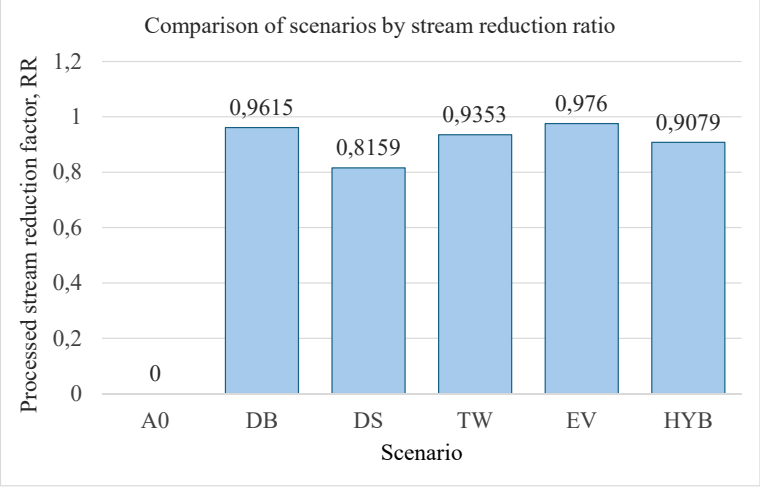


Fig. 2. Comparison of scenarios by stream reduction ratio

Table 2

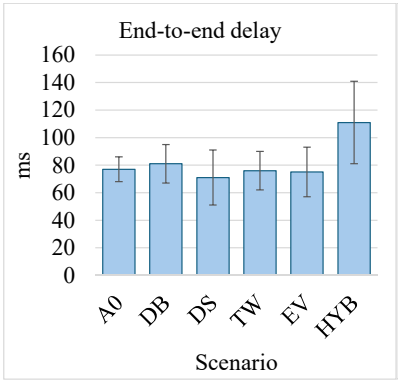
Results of stream reduction assessment after edge processing

Scenario	Parameter	N_{proc} over 600 s	R_{proc}, 1/s	RR_{proc}
A0	–	597.33 ± 0.94	0.9956 ± 0.0016	0
DB	$dT = 0.2$	23.00±11.43	0.0383 ± 0.019	0.9615
	$dH = 1$			
DS	$T_{out} = 5$s	110.00 ± 64.15	0.1833 ± 0.107	0.8159
TW	$W = 30$s	38.67 ± 0.94	0.0644 ± 0.0016	0.9353
	mean			
EV	$T_{high} = 28$	14.33 ± 13.02	0.0239 ± 0.022	0.9760
	$H_{high} = 70$			
HYB	$dT = 0.2$	55.00 ± 1.41	0.0917 ± 0.002	0.9079
	$dH = 1$			
	$hb = 30$s			

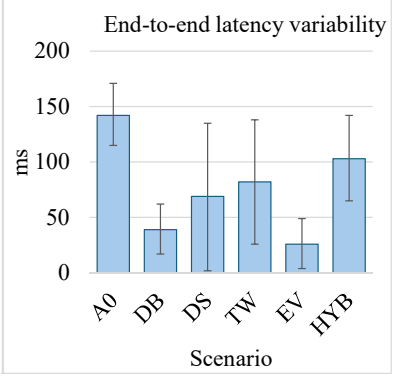
The time characteristics of the path "edge level → SCADA → database" also revealed a dependence on the type of scenario. For most scenarios, the average end-to-end latency Δt_{e2e} was in a close range: 77.03 ± 8.96 ms for A0, 81.20 ± 13.44 ms for DB, 71.06 ± 20.11 ms for DS, 75.99 ± 13.87 ms for TW, and 75.55 ± 18.64 ms for EV. The largest value was recorded for HYB – 112.46 ± 30.35 ms. A graphical comparison of the average end-to-end latency is shown in Fig. 3, *a*.

For individual sections of the path, the largest difference was observed in the SCADA → DATABASE segment. For HYB, the average Δt_{db} value was 62.93 ± 26.74 ms, while for the other scenarios this indicator was approximately in the range of 17.70–26.35 ms. Therefore, in terms of Δt_{db}, the HYB scenario had the highest value among the studied scenarios.

The jitter indicator also demonstrated scenario dependence. For the base scenario, its value was 142.32 ± 27.27 ms. Lower values were observed for DB and EV – 38.52 ± 22.23 ms and 25.98 ± 22.83 ms, respectively. For DS, the jitter was 68.73 ± 66.37 ms, for TW – 81.83 ± 55.93 ms, and for HYB – 103.01 ± 39.54 ms. A graphical comparison of the jitter values for the studied scenarios is shown in Fig. 3*b*. Therefore, edge optimization affected not only the intensity of the processed stream but also the stability of the time characteristics of the path. Table 3 shows a comparison of the time metrics for the path "edge level → SCADA → database" in the studied scenarios in the form of $\mu \pm \sigma$.



a



b

Fig. 3. Time characteristics of the path:
a – end-to-end latency value;
b – end-to-end jitter value

Table 3

End-to-end latency and jitter values for the scenarios studied

Scenario	End-to-end latency, ms ($\mu \pm \sigma$)	End-to-end jitter, ms ($\mu \pm \sigma$)
A0	77.03 ± 8.96	142.32 ± 27.27
DB	81.20 ± 13.44	38.52 ± 22.23
DS	71.06 ± 20.11	68.73 ± 66.37
TW	75.99 ± 13.87	81.83 ± 55.93
EV	75.55 ± 18.64	25.98 ± 22.83
HYB	112.46 ± 30.35	103.01 ± 39.54

The data above show that the lowest average end-to-end latency was recorded for the DS scenario, while the highest was recorded for HYB. According to the jitter metric, the lowest average was recorded for EV, and the highest was recorded for A0.

5. 4. Results of evaluating the resource profile of system nodes

The system resource profile was evaluated using Prometheus telemetry for edge, SCADA, and DATABASE nodes; measurement results are given in Table 4. For the edge node, the average CPU load values in all scenarios remained close and ranged from 1.22 ± 0.10% to 1.34 ± 0.09%. The use of RAM also changed slightly – from 1.06 ± 0.15 GiB to 1.27 ± 0.00 GiB. That showed that under the conditions of this bench, the applied edge operators did not generate a significant additional computational load at the edge level.

Table 4

Resource metrics of system nodes under different streaming scenarios

Scenario	Edge CPU, %	Edge RAM, GiB	Edge NET RX, KiB/s	Edge NET TX, KiB/s	SCADA CPU, %	SCADA RAM, GiB	SCADA NET RX, KiB/s	SCADA NET TX, KiB/s	DB CPU, %	DB RAM, GiB
A0	1.34 ± 0.09	1.27 ± 0.00	3.48 ± 0.05	4.15 ± 0.01	3.23 ± 0.65	18.41 ± 0.34	13.35 ± 0.39	5.66 ± 0.53	35.90 ± 0.26	8.42 ± 0.02
DB	1.33 ± 0.02	1.27 ± 0.00	3.54 ± 0.12	2.15 ± 0.05	3.68 ± 1.45	17.10 ± 0.31	69.00 ± 82.74	20.02 ± 22.32	34.70 ± 0.74	8.42 ± 0.02
DS	1.26 ± 0.06	1.16 ± 0.15	3.15 ± 0.57	2.13 ± 0.46	3.39 ± 0.34	17.53 ± 1.69	200.70 ± 258.96	9.44 ± 7.72	25.67 ± 12.46	7.24 ± 1.72
TW	1.22 ± 0.10	1.17 ± 0.15	3.45 ± 0.06	1.24 ± 0.06	3.64 ± 1.45	18.03 ± 1.99	52.26 ± 34.16	7.24 ± 2.41	23.75 ± 14.86	7.33 ± 1.75
EV	1.25 ± 0.02	1.06 ± 0.15	3.43 ± 0.14	1.05 ± 0.02	2.92 ± 1.04	17.24 ± 2.29	81.28 ± 97.09	7.27 ± 4.65	13.71 ± 15.37	6.04 ± 1.82
HYB	1.25 ± 0.08	1.27 ± 0.00	3.67 ± 0.04	1.29 ± 0.01	3.91 ± 1.11	19.33 ± 0.20	27.39 ± 16.02	6.94 ± 1.00	34.97 ± 0.88	8.64 ± 0.01

The most pronounced resource effect was observed for network traffic at the output of the edge node. For the baseline scenario A0, the average value of Edge NET TX was 4.15 ± 0.01 KiB/s. In the optimization scenarios, it was lower: 2.15 ± 0.05 KiB/s for DB, 2.13 ± 0.46 KiB/s for DS, 1.24 ± 0.06 KiB/s for TW, 1.05 ± 0.02 KiB/s for EV, and 1.29 ± 0.01 KiB/s for HYB. A graphical comparison of the scenarios by network traffic at the output of the edge node is shown in Fig. 4. Thus, a decrease in the intensity of the processed stream was accompanied by a decrease in the network load at the output of the edge node.

For SCADA and DATABASE nodes, resource indicators were characterized by greater inter-measurement variability. At the SCADA level, the average CPU utilization ranged from 2.92 ± 1.04% to 3.91 ± 1.11%, and the RAM usage ranged from 17.10 ± 0.31 GiB to 19.33 ± 0.20 GiB. For the DATABASE node, the average CPU utilization ranged from 13.71 ± 15.37% to 35.90 ± 0.26%, and the RAM usage ranged from 6.04 ± 1.82 GiB to 8.64 ± 0.01 GiB. The available telemetry set for the DATABASE node lacked network and disk metrics, so the evaluation of the resource profile of this node was limited to CPU and RAM.

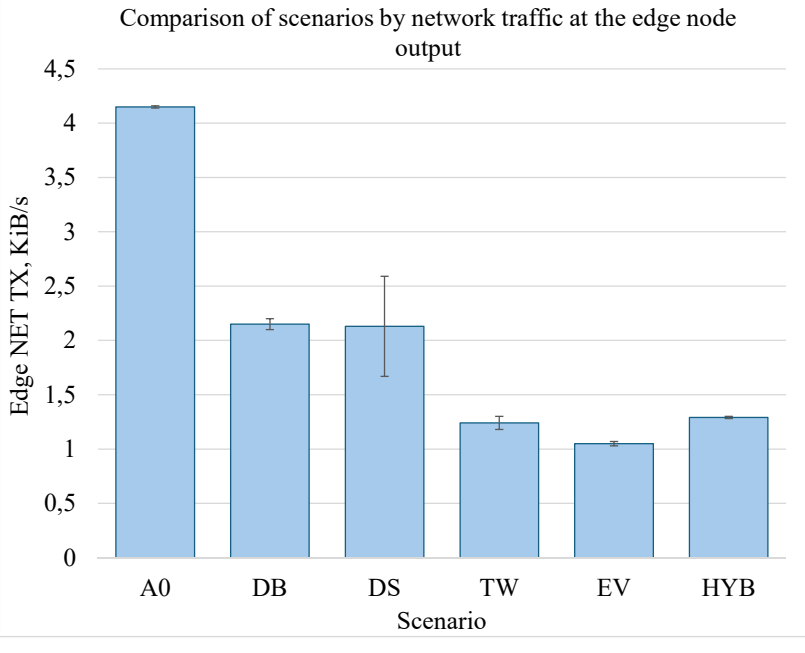


Fig. 4. Comparison of scenarios by network traffic at the edge node output

Our results show that within the studied configuration, edge stream processing changed primarily the network component of the load, while the computational load on the edge node remained low.

6. Discussion of results based on investigating edge stream processing of process data

Our results show that edge stream processing of process data changes not only the number of records transmitted to the SCADA system and database but also the time and resource characteristics of the transmission chain. In all the studied scenarios, the intensity of the processed stream was lower than under the basic mode A0. At the same time, different scenarios had different effects on the stream reduction, end-to-end latency, jitter, and network load at the output of the edge node.

These results build on the approach proposed in previous work [11], in which a theoretical model of stream processing of process data in the chain "data source → SCADA → database" was built. In this paper, that approach is extended to the edge level and was tested for several processing scenarios taking into account stream reduction, latency, jitter, and load on the system nodes. This is consistent with approaches to moving some of the processing closer to the data sources in IIoT and edge computing environments [7, 8].

Fig. 2 and Table 2 demonstrate that the use of edge operators in all scenarios reduced the intensity of the processed stream compared to the base mode A0. The greatest reduction in stream was obtained for the EV and DB scenarios, for which the reduction coefficient was 0.9760 and 0.9615, respectively. This is explained by the principle of operation of these operators: in the EV and DB scenarios, data is transmitted only when a given event or a significant change in the parameter occurs, so minor signal fluctuations do not form new records. The DS scenario, on the contrary, performs periodic sampling regardless of the nature of the signal change, so the stream remains more regular, but less reduced. The TW scenario occupies an intermediate position because aggregation in the time window reduces the number of messages but does not perform event or threshold data clipping.

According to Fig. 3 and Table 3, a decrease in stream intensity did not always have the same effect on the time characteristics of the path. For most scenarios, the average end-to-end latency remained within the range: from 71.06 ms for DS to 81.20 ms for DB. The HYB scenario stands out, for which the latency increased to 112.46 ms. This shows that stream reduction alone does not guarantee a better system time profile; latency is affected

not only by the number of records transferred but also by how the processed stream is formed.

The result for the SCADA → database segment is indicative: the HYB scenario demonstrated the highest value of $\Delta t_{db} = 62.93$ ms, while for other scenarios this indicator ranged from 17.70 to 26.35 ms. The hybrid mode combines deadband filtering and periodic state updates, which forms less uniform inter-message intervals compared to scenarios with purely threshold or event selection. The end-to-end latency for the kth record can be given as $\Delta t_{e2e,k} = \Delta t_{es,k} + \Delta t_{db,k}$, where $\Delta t_{es,k}$ is the latency in the "edge node → SCADA" section, and $\Delta t_{db,k}$ is the latency in the "SCADA → database" section. Therefore, the increased variability of the $\Delta t_{db,k}$ component, interpreted here as latency jitter, may be one of the factors contributing to the higher end-to-end jitter observed in the HYB scenario.

The significant relative spread in the EV scenario can be explained by the event-based nature of the formation of the processed stream. In some measurements, the condition was rarely triggered, so the number of records was small. In other measurements, events occurred in series, which gave greater inter-measurement variability.

For the DS scenario, such a spread was less expected because periodic sampling, by its logic, should form a more stable stream. In this experiment, the deviations are probably related to the peculiarities of the implementation of periodic sampling, time synchronization, and the passage of messages through the edge, SCADA, and DATABASE levels. Therefore, it is advisable to consider this result as a limitation of the experiment, which requires separate verification in further studies.

Additionally, our results show that for industrial automation systems, not only the average latency is important but also jitter. This is consistent with studies of IIoT protocols, in which latency and jitter are considered key characteristics of the transmission of high-frequency industrial flows [9]. A scenario can have an acceptable average latency value but, at the same time, operate unstable over time. For monitoring, signaling, and decision support tasks, this is important because the system must not only transmit data quickly but also do it predictably.

According to Fig. 3, *b*, and Table 3, in the baseline scenario, the value of the jitter was 142.32 ms. For DB and EV, this figure was significantly lower: 38.52 ms and 25.98 ms, respectively. This means that event and threshold selection not only reduced the stream intensity but also formed a more stable time profile of data delivery. Therefore, scenarios with less jitter are more practical than scenarios in which the average latency remains acceptable, but its spread is significant.

According to Table 4, the average CPU utilization on the edge node in all scenarios remained almost at the same level and was within 1.22–1.34%. RAM utilization also changed slightly. The results show that for this scale of the experiment, the preprocessing operators did not overload the local edge node and remained acceptable in terms of resource costs.

Fig. 4 and Table 4 demonstrate that the most noticeable change in resource performance was related to network traffic at the output of the edge node. In the baseline scenario A0, the Edge NET TX value was 4.15 KiB/s. After applying edge processing, it decreased to 1.05 KiB/s for EV, to 1.24 KiB/s for TW, and to 1.29 KiB/s for HYB.

This shows that the reduction of the processed stream directly reduced the network load at the output of the edge node. This is important for industrial automation systems because such optimization reduces not only the number of records in the database but also the amount of traffic between the layers of the architecture. That is, the effect of edge processing is manifested not only in the database but also in the load on the communication infrastructure.

Our results are consistent with the general provisions of stream processing in which a continuous stream of events is considered as an object of processing in real or close to real time, taking into account temporal semantics, state, and window operations [3]. In this work, this approach is tested not for a general stream of events but for process data in the industrial chain "field-level equipment → edge node → SCADA → database". For such a chain, not only stream reduction is important but also latency, jitter, and resource load on system nodes.

Unlike papers [3, 4], which focus on general models and performance of streaming systems, in this study the evaluation was performed for the applied industrial chain "field-level equipment → edge node → SCADA → database". In contrast to [5, 6], which consider the general organization of industrial data processing and Edge-Fog-Cloud architecture, our results make it possible to compare specific edge operators in terms of stream reduction, latency, jitter, and resource load under the same experimental conditions. This became possible by defining the processed stream as a separate object of evaluation and the use of a single transport path for all scenarios.

In the work, five typical scenarios of edge stream processing were quantitatively compared in terms of functional, time, and resource indicators. The results are not limited to confirming the fact of reducing the stream intensity after filtering. It is shown that different processing scenarios have different effects on stream reduction, end-to-end latency, jitter, and resource load on system nodes.

The scenarios were compared by the degree of stream reduction, by latency, jitter, and resource profile of the system nodes. In practice, EV and DB can be a guideline for the tasks of maximum stream reduction, DS – for modes with controlled transmission periodicity, and HYB – for cases where a guaranteed periodic state update is required.

Our methodology has several limitations. First, the experiment was performed on a specific bench with a fixed topology and software stack, so the absolute values of latencies and load should not be directly generalized to all industrial systems. Second, the stream source was implemented on the basis of a specific sensor and a specific message generation mode. Third, some resource indicators were missing for the DB server in telemetry. At the same time, the proposed approach is reproducible and suitable for comparing edge stream processing strategies under controlled conditions.

The results obtained have a number of limitations that must be taken into account during their practical use and further theoretical generalization. Our study was performed on a laboratory bench, and not at a real industrial facility. Therefore, some of the time and resource characteristics may differ under production operation conditions. The experiment covers a limited set of scenarios and one type of logic of streaming operators; because of this, the derived dependences should not be mechanically transferred to all possible classes of stream processing. The available set of telemetry for the DATABASE node lacked network and disk metrics. Therefore, the assessment of the resource profile of this level remains incomplete. The study was performed with a fixed bench configuration and does not cover scaling with an increase in the number of sources, stream intensity, or complexity of processing rules.

The shortcomings of the study include the lack of testing scenarios with variable input stream intensity, different types of process signals, and a larger number of simultaneous data sources. The work did not perform parametric optimization of thresholds, time windows, and heartbeat interval, so the obtained values characterize the selected test settings, and not the optimal modes for all possible industrial applications.

Further research should be directed at testing the results on more complex benches that will be closer to actual production conditions. This is important as the heterogeneity of network conditions, an increase in the number of sources, and a higher intensity of flows can change the relationship between reduction, latency, and resource load. Of separate consideration is to expand the set of operators, in particular due to more complex event processing rules, combined window schemes, and multi-channel scenarios. Another promising direction is the combination of experimental evaluation of edge processing scenarios with the analysis of information entropy and spectral characteristics of process signals, which could allow us to assess not only the reduction in the number of records and the load on the infrastructure but the degree of preservation of informationally significant changes in the primary signal after reduction as well.

It is also advisable to deepen the analysis of temporal predictability, in particular the relationship between the type of processed stream, jitter, and the suitability of a scenario for different classes of industrial tasks. Another area is related to the integration of stream processing with analytical and intelligent modules, in particular for detecting anomalies, predicting equipment status, and supporting decision-making at the upper levels of the architecture.

7. Conclusions

1. We have proposed a generalized scheme for data stream and processing in a multi-level industrial system "field-level equipment → edge node → SCADA → database" and justified the feasibility of performing some operators at the edge level. That has made it possible to link the functional results from processing with changes in stream intensity, time characteristics of the path, and the resource profile of system nodes within the adopted experimental configuration.

2. An experimental bench was implemented; six edge processing scenarios were formed, representing different classes of stream operators: basic mode without optimization, deadband filtering, periodic discretization, time aggregation, event filtering, and a hybrid scheme. That provided the same conditions for comparative evaluation of scenarios according to the criteria of reduction of the processed stream, end-to-end latency, jitter, and resource indicators of system nodes.

3. It was found that all the studied scenarios of edge stream processing reduce the intensity of the processed stream relative to the baseline mode. The greatest reduction was provided by EV ($RR_{proc} = 0.9760$) and DB ($RR_{proc} = 0.9615$), while DS ($RR_{proc} = 0.8159$) formed a less reduced, but more regular stream. For most scenarios, the average end-to-end latency remained in the close range of 71.06–81.20 ms, while for HYB it increased to 112.46 ms. The lowest values of jitter were recorded for EV (25.98 ms) and DB (38.52 ms), which indicates differences between the scenarios not only in the degree of stream reduction but also in the temporal stability of their operation.

4. Within the experimental bench, the resource profile of the system nodes has been estimated, which, in this work, was defined as a set of measured indicators of CPU, RAM, and network traffic usage for individual nodes of the experimental configuration. For the edge and SCADA nodes, the resource profile covered CPU, RAM, NET RX, and NET TX, while for the DB node only CPU and RAM were taken into account because network and disk metrics were not available for it in the existing telemetry set. For the edge node, the average CPU usage in all scenarios remained within 1.22–1.34%, and the RAM usage changed slightly. The most pronounced changes were observed for the network traffic at the edge node output: Edge NET TX decreased from 4.15 KiB/s in the baseline scenario to 2.15 KiB/s for DB, 2.13 KiB/s for DS, 1.24 KiB/s for TW, 1.05 KiB/s for EV, and 1.29 KiB/s for HYB. Thus, in the adopted experimental configuration, the edge stream scenarios primarily affected the network component of the resource profile, while the computational load of the edge node remained almost unchanged.

Conflicts of interest

The authors declare that they have no conflicts of interest in relation to the current study, including financial, personal, authorship, or any other, that could affect the study and the results reported in this paper.

Funding

The study was conducted without financial support.

Data availability

The data will be provided upon reasonable request.

Use of artificial intelligence

The authors used ChatGPT, the GPT-5.5 Thinking model, in the study, within the limits permitted by the journal's policy. The tool was used to search for sources by keywords and criteria specified by the authors. To pre-process possible methodological approaches, check grammar, spelling, and punctuation without changing the content of the manuscript. To create a graphic annotation based on the original author data.

All sources were checked by the authors for bibliographic data, DOI, and relevance to the research topic. Experimental data, calculations, tables, figures, research results, and conclusions were obtained and checked by the authors independently. AI was not used to generate experimental data, write manuscript sections, describe research results, or draw conclusions in the manuscript.

Authors' contributions

Igor Krasnikov: Conceptualization, Methodology, Supervision, Project administration, Writing – review & editing; **Kyrylo Halliamov**: Conceptualization, Methodology, Investigation, Software, Data curation, Formal analysis, Visualization, Writing – original draft; **Ihor Lysachenko**: Software, Validation, Resources, Writing – review & editing.

References

1. Wang, S., Wan, J., Zhang, D., Li, D., Zhang, C. (2016). Towards smart factory for industry 4.0: a self-organized multi-agent system with big data based feedback and coordination. Computer Networks, 101, 158–168. https://doi.org/10.1016/j.comnet.2015.12.017
2. Farooq, M. S., Abdullah, M., Riaz, S., Alvi, A., Rustam, F., Flores, M. A. L. et al. (2023). A Survey on the Role of Industrial IoT in Manufacturing for Implementation of Smart Industry. Sensors, 23 (21), 8958. https://doi.org/10.3390/s23218958
3. Fragkoulis, M., Carbone, P., Kalavri, V., Katsifodimos, A. (2023). A survey on the evolution of stream processing systems. The VLDB Journal, 33 (2), 507–541. https://doi.org/10.1007/s00778-023-00819-8
4. Vikash, Mishra, L., Varma, S. (2020). Performance evaluation of real-time stream processing systems for Internet of Things applications. Future Generation Computer Systems, 113, 207–217. https://doi.org/10.1016/j.future.2020.07.012
5. Kammerer, K., Pryss, R., Hoppenstedt, B., Sommer, K., Reichert, M. (2020). Process-Driven and Flow-Based Processing of Industrial Sensor Data. Sensors, 20 (18), 5245. https://doi.org/10.3390/s20185245
6. Kumar, R., Agrawal, N. (2023). Analysis of multi-dimensional Industrial IoT (IIoT) data in Edge–Fog–Cloud based architectural frameworks: A survey on current state and research challenges. Journal of Industrial Information Integration, 35, 100504. https://doi.org/10.1016/j.jii.2023.100504
7. Qiu, T., Chi, J., Zhou, X., Ning, Z., Atiquzzaman, M., Wu, D. O. (2020). Edge Computing in Industrial Internet of Things: Architecture, Advances and Challenges. IEEE Communications Surveys & Tutorials, 22 (4), 2462–2488. https://doi.org/10.1109/comst.2020.3009103
8. Xhafa, F., Kilic, B., Krause, P. (2020). Evaluation of IoT stream processing at edge computing layer for semantic data enrichment. Future Generation Computer Systems, 105, 730–736. https://doi.org/10.1016/j.future.2019.12.031
9. Sarasola, T. F. D. B., García, A., Ferrando, J. L. (2024). IIoT Protocols for Edge/Fog and Cloud Computing in Industrial AI. International Journal of Cloud Applications and Computing, 14 (1), 1–30. https://doi.org/10.4018/ijcac.342128
10. Lu, D., Wang, K., Wang, Y., Shen, Y. (2025). The Proposal and Validation of a Distributed Real-Time Data Management Framework Based on Edge Computing with OPC Unified Architecture and Kafka. Applied Sciences, 15 (12), 6862. https://doi.org/10.3390/app15126862
11. Krasnikov, I., Hallyamov, K. (2026). Theoretical Foundations of Streaming Processing of Technological Data in Industrial Automation Systems. Bulletin of the National Technical University “KhPI” A Series of “Information and Modeling”, 1 (1 (15)), 69–82. https://doi.org/10.20998/2411-0558.2026.01.04