Data Collection and Processing Method in the Networks of Industrial IOT

Larysa Globa, Vasyl Kurdecha(✉), Demyd Popenko, Maksym Bezvuhliak, and Yevgeniy Porolo

National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine
lgloba@its.kpi.ua, kvv.vasyl@gmail.com

Abstract. The implementation of IIoT allows users to use data and analytics for predictive analysis, reduced equipment downtime, centralized storage and remote asset monitoring. With the introduction of reasonable production information obtained from the devices must be processed that requires significant time and memory to store the accumulated information. This poses the task of minimizing processing time and file size for network transmission.

The main purpose of this work is improving the efficiency of information collection and processing in the industrial Internet of Things through the use of the modified method.

The method of serialization and deserialization of data on the basis of the Protobuf method is offered in the work, which allows to increase the efficiency of information processing by time indicators and to reduce the volume of transmitted information. In the process of work achieved an increase in processing speed information received from IIoT equipment through use serialization mechanism, and reduce the file size of the information that transmitted over the network.

Keywords: IIoT · Collection of information · Processing of information · Serialization · Protobuf

1 Introduction

There are three existing methods [1] of data collection and processing, which were analyzed: "Method of collecting and processing information using a gateway", "Method of collecting and processing information using a smart hub", "Method of collecting and processing information based on industrial buses". The method of collecting and processing information using a smart hub has an advantage over other methods because it helps to modernize and achieve a high level of production in the manufacturing industry.

Devices and sensors generate and collect a huge amount of data, the total amount of data collected may be so large that it may be impossible to transmit them over the network for further analysis. Sensors can transmit this information every 30 s, and there may be several hundred such devices in stock. It can be just one of dozens of types of sensors that have different types of data, which together represent a massive inhomogeneous

D. Perakovic and L. Knapcikova (Eds.): FABULOUS 2022, LNICST 445, pp. 153–167, 2022.
https://doi.org/10.1007/978-3-031-15101-9_11

set of information from many sources. This leads to high costs for data collection and integration of equipment with the data processing system. Also a big problem is the lack of computing power and storage resources to perform complex tasks of analysis and machine learning.

In addition, large amounts of data are difficult to process because more time is required to respond. Thus, the transmission of confidential data over the Internet to perform important analysis often becomes a problem [2].

2 Background

A. The method of collecting and processing information using a gateway

This method contains three modules: the intelligent object module, the gateway module, and the control center (server) module. Each module is multilevel (including touch, network and application levels) and performs certain functions to support the monitoring of the interdependent environment [3].

The gateway module is a bridge between smart objects and the control center. On the other hand, in the case where intelligent objects do not have a direct connection available for direct communication with the control center via telecommunications technology, the gateways will connect these smart objects with the control center. Also, the gateway module will perform the duties of the application layer in the absence of the application layer in the module of the smart object.

B. A method of collecting and processing information using a smart hub

A method of collecting and processing information that allows the use of information and operational infrastructure to improve the management, monitoring and control of existing and new devices. It consists of three main components: a layer of devices, an intelligent hub, a microservice cloud platform. The presented method can also be implemented in existing intelligent enterprises to optimize the management of intelligent equipment and to increase production efficiency [4].

The intelligent hub can act as a gateway for existing IIoT devices. Functionally, the hub offers a simple and convenient way to pre-process data, exchange data and communicate between existing IIoT devices and the server platform, which means that using the intelligent hub can update or improve data exchange and communication channels.

C. Method of collecting and processing information based on industrial tires

This method is performed on four layers: the layer of devices, sensors and mechanisms, the layer of the data provider, the layer of cross-platform software and the application layer [5].

The data provider layer is designed to receive data from industrial buses and transfer them to the level of cross-platform software, which contains a software communication module for each industrial bus.

The purpose of this module is to implement an information collection cycle. The interface for receiving and transmitting data is used by the data provider, a separate software module, to receive data from industrial networks and to store them in the buffer cache. This memory is used to respond more quickly to requests from the middleware level.

The method of collecting and processing information using a smart hub has an advantage over other methods because it helps in modernizing and achieving a high level of production in the manufacturing industry. This is achieved due to the fact that the main element of this method is an intelligent hub, which provides convenient solutions for scaling the system with new devices, performs data collection and formatting using serialization processes, which reduces the amount of information needed to be transferred to the cloud platform for further analysis.

3 Modified Method of Collection and Processing Data

Based on the selected prototype, the modified method of collecting and processing information is as follows (Fig. 1).

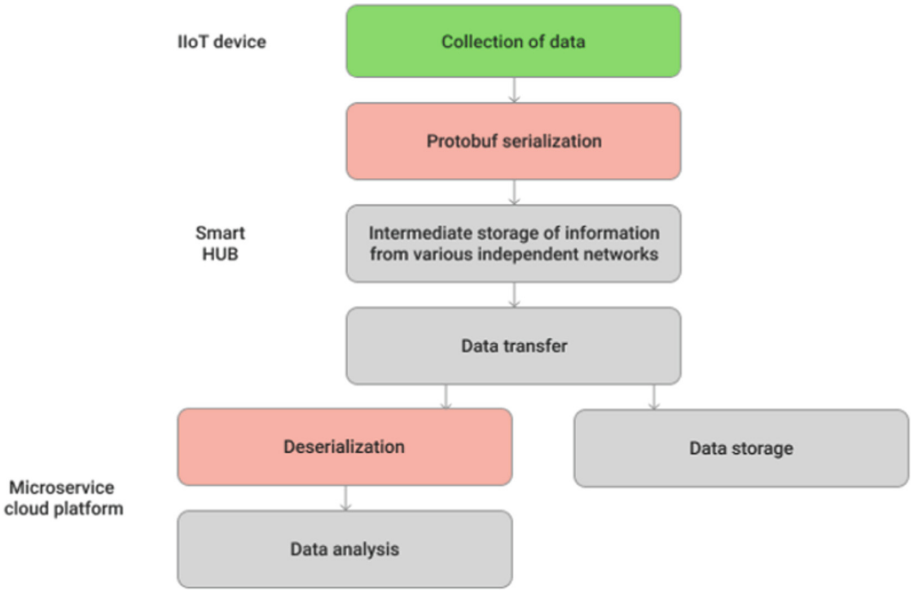

Fig. 1. Modified method

The basis of data collection are sensors, mechanisms, which are scattered in a certain geographical area. They exchange data over the network for autonomous data collection and transmission to a special node, which is considered an intermediate point for collecting information.

To store and transmit the information received from the devices, a serialization mechanism is used, which allows to reduce the size of the transmitted files, which in turn reduces the load on the network and reduces the time required to transmit information for further analysis.

The information received from sensors, mechanisms, devices is transferred to the intelligent hub, using the Internet of Things protocol CoAP or MQTT.

Analyzed on the IIoT microservice platform, the information can be transferred to other information or operating systems or stored in a database.

A. Collecting data

Depending on the requirements for information collection, the transfer of collected data may be carried out periodically or on an event basis. An intelligent hub is a node with two or more network interfaces, it collects and performs intermediate processing of data received from information collection devices.

Sensorsand mechanisms are small electronic devices capable of measuring physical quantities (eg, temperature, light, pressure) and transmit it to the information processing unit. Given the achievements in the field of microelectronics, technology and software for wireless data transmission allow to produce microsensors with a volume of several cubic millimeters [5].

The general process of collecting information and transmitting it to an intelligent hub is shown in Fig. 2. Devices can have their own communication methods, such as Wi-Fi, BLE, ZigBee and Z-Wave.

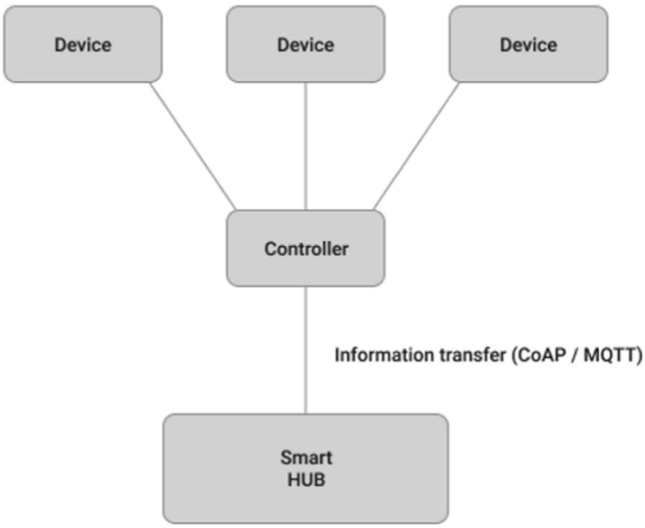

Fig. 2. The general process of collecting information

B. Technologies transmition data to HUB

Sensor networks are spatially distributed sensors that monitor physical or environmental conditions, such as temperature, sound, pressure, etc., and jointly transmit their data over the network. Such networks can be used in a large number of different applications: industrial automation, microclimate control systems, security and fire alarm systems, energy consumption accounting and optimization, etc. [6]. The coverage area of such networks can be from several meters to several kilometers. One of the main standards for the implementation of these networks are Wi-Fi, BLE, ZigBee, Z-Wave.

C. Protocols for information data

In the modified method, the CoAP protocol is intended for transmission between devices in one limited network, for example, with low power consumption, in lossy networks, as well as between devices in different networks connected to the Internet. CoAP can be integrated with data formats such as XML, JSON, Protobuf for effective communication with other platforms.

In the modified method, the MQTT protocol provides minimum resource requirements and is used to transmit information. The protocol does not impose restrictions on the data format.

D. Serialization

Figure 3 shows the general process of serialization-deserialization.

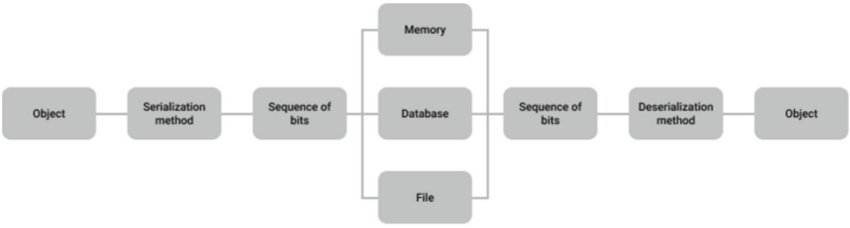

Fig. 3. The general process of serialization-deserialization

In the proposed method of creating a distributed service, parts of which must exchange information with a complex structure, in this case, for the data to be transmitted, a code is created that performs the processes of serialization and deserialization. For the object filled with the necessary data, the created serialization code is called, as a result on an output we receive, for example, an XML file.

The resulting sequence of bits is written to a database, memory or file, which is then sent to the receiving party. For deserialization, the recipient service creates an object of the same type and calls the required code, resulting in an object with the same data as the sender service object.

Comparison of serialization methods

Table 1 presents a description of the main characteristics of serialization methods using XML, JSON, Protobuf.

For a modified method of collecting and processing information, it is proposed to use the method of serialization Protobuf.

Compared to Protobuf, JSON and XML pass metadata details, which adds a payload to the payload. Using Protobuf for serialization and deserialization will consume less CPU time and memory, so processing time is faster compared to JSON and XML.

Protobuf compresses data and generates dense data. When compared to XML, it is almost 1/3 of the size, and when compared to JSON, then 1/2 [7].

JSON and XML are readable by humans and are not secure for data transmission over the network. If it is necessary that the answer is not read by the user, then it is necessary to use Protobuf. The user also needs an proto file to deserialize the object stream.

E. Microservice platform

In the modified method, the microservice platform structures the application as a set of services. Microservices are usually based on business functions.

Deployment using containers provides portability. Containerization reduces deployment time, as it requires only the inclusion of the required container, without affecting other containers running on the same host [8].

Table 1. Description of the main characteristics of serialization methods

Features	JSON	XML	ProtoBuf
Standardization	Yes	Yes	Yes
Specification	STD 90/RFC 8259	Recommendations W3C: 1.0 and 1.1	Developer Guide: Encoding
Binary	No	Partly	Yes
Available for human reading	Yes	Yes	No
Link support	Yes	Yes	No
Standard APIs	JSONQuery, JSONPath, JSON-LD	DOM, SAX, XQuery, XPath	C++, Java, C#, Python, and other

The microservice platform may transmit useful statistics or information to other information or operating systems, such as a production management system, an automated storage and retrieval system, and an enterprise asset management system using Protobuf using the CoAP or MQTT protocol.

4 Comparative Evaluation of the Proposed Method

The simulation process serializes the files before sending them over the Internet, as this can significantly reduce the time required to do so. Also serialized files reduce the amount of space required for data storage [9, 10].

To obtain a comparative assessment of the proposed solution, three methods of serialization using XML, JSON, Protobuf were studied according to the following criteria (Fig. 4):

– the time required to serialize the input data;
– time required to deserialize serialized data;
– time required for serialization and deserialization of input data;
– the size of the serialized file.

According to the selected criteria, it is possible to determine how much the time and speed of information processing required to send information will be reduced, using the proposed method.

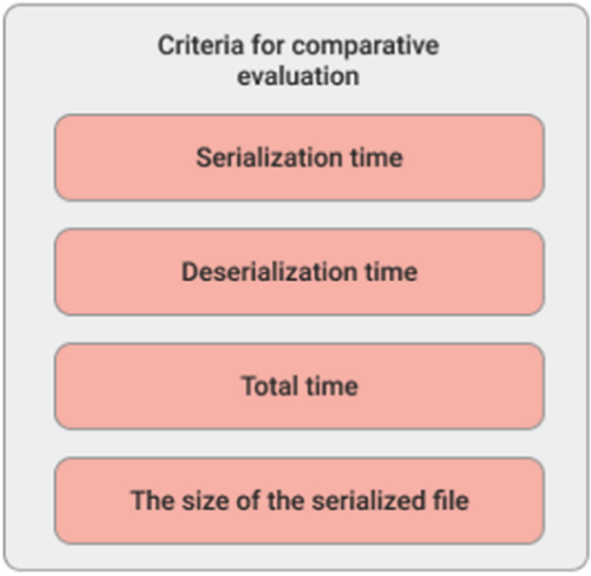

Fig. 4. Criteria for comparative evaluation

Figure 5 shows a full-scale model. The information generated by the device is transmitted to the controller. The process of serialization is called on the controller, as a result of which we get a serialized file that can be transmitted over the network. The deserialization process is called on the receiver side and the initial information is restored.

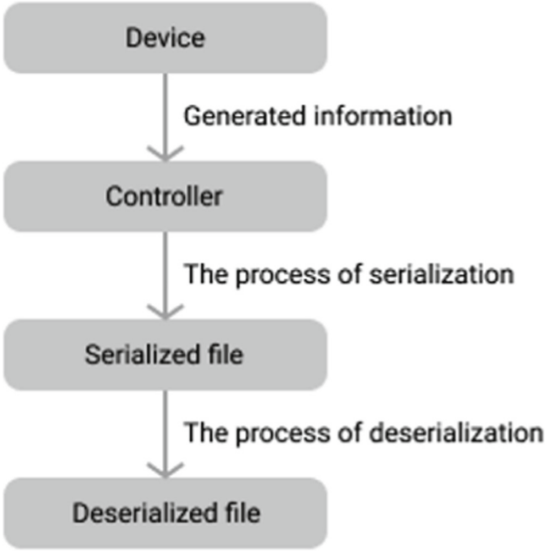

Fig. 5. Full-scale model

The tests are repeated for different amounts of input data (Table 2). Initialize the experiment to write to the file by different methods of serialization of 50, 500, 5000, 5000 lines of the input file. In each test, write the appropriate number of rows of generated data to a file. For each method (XML, JSON, Protobuf) we perform serialization and deserialization operations 9 times and calculate the average value for each method. In each of the 9 iterations for each method, we measure the time of serialization, deserialization, and the total time in milliseconds. We also measure the size of the serialized file for each method. Figure 6 shows the process of modeling.

Using the method of finding intermediate values for a known discrete set of values, we determine:

1. For serialization time, deserialization time and total time

$$t = b + \left(\frac{n - n_0}{n_1 - n_0}\right) * (a - b),\ n_0 \le n < n_1 \tag{1}$$

where

t is the serialization/deserialization/total time,

n is the number of lines of the input file,

n_0 is the initial number of lines of the input file,

n_1 is the maximum number of lines of the input file,

a is the time of serialization/deserialization/total time with the maximum number of lines of the input file,

b is the serialization/deserialization/total time at the initial number of lines of the input file.

2. For the size of the serialized file

$$s = d + \left(\frac{n - n_0}{n_1 - n_0}\right) * (c - d),\ n_0 \le n < n_1 \tag{2}$$

where

s is the size of the serialized file,

n is the number of lines of the input file,

n_0 is the initial number of lines of the input file,

n_1 is the maximum number of lines of the input file,

c is the size of the serialized file with the maximum number of lines of the input file,

d is the size of the serialized file lines of the input file. For the size of the serialized file.

Table 2. Input parameters for simulation

Data types	int, double, string, list of objects
Number of input data (lines)	50, 500, 5000, 50000
Number of tests	4
Number of iterations for each test	9

The obtained mathematical expressions will allow you to quickly calculate the time of serialization, deserialization and total time, as well as the size of the serialized file depending on the amount of input data.

For serialization in tests the class Object which consists of types int, double, string, and also the list of objects is used. The test results are stored in text files.

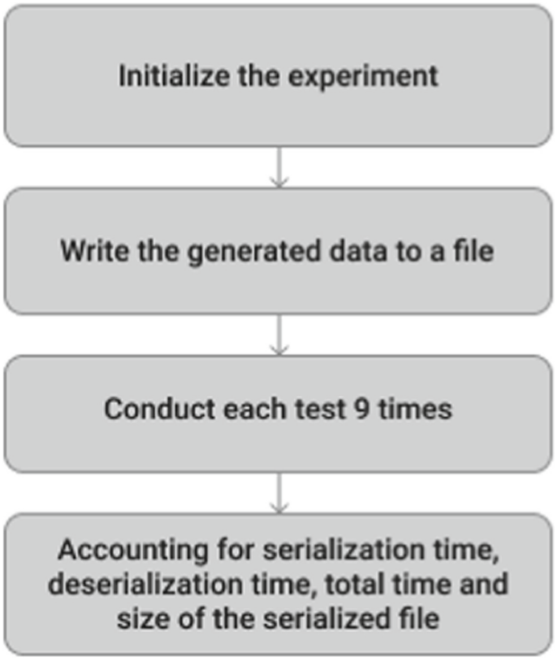

Fig. 6. Modeling process

The results will always be different for different hardware, operating systems, but there is one thing that remains the same - the relative difference between the methods of serialization.

Evaluation of the proposed method of serialization

To conduct a comparative assessment, we will build graphs according to certain criteria (the time required to serialize the input data).

According to the results of four tests, Table 3 shows the generalized results of the serialization time for the three methods. Figure 7 shows a graph of serialization time depending on the number of lines of the input file.

According to the simulation results, the following mathematical expression for the serialization time using Protobuf was obtained:

$$t = \begin{cases} 0{,}508 * n, 0 \leq n < 50 \\ 25{,}4 - 1{,}91 * 10^{-2} * (n - 50), 50 \leq n < 500 \\ 16{,}8 + 4{,}467 * 10^{-3} * (n - 500), 500 \leq n < 5000 \\ 36{,}9 + 4{,}311 * 10^{-3} * (n - 5000), 5000 \leq n < 50000 \end{cases}$$

Table 3. Input serialization time

The number of lines of the input file	Protobuf (ms)	JSON (ms)	XML (ms)
50	25,4	37,1	25,6
500	16,8	24,7	33,2
5000	36,9	100,3	285,7
50000	230,9	753,4	2594,2

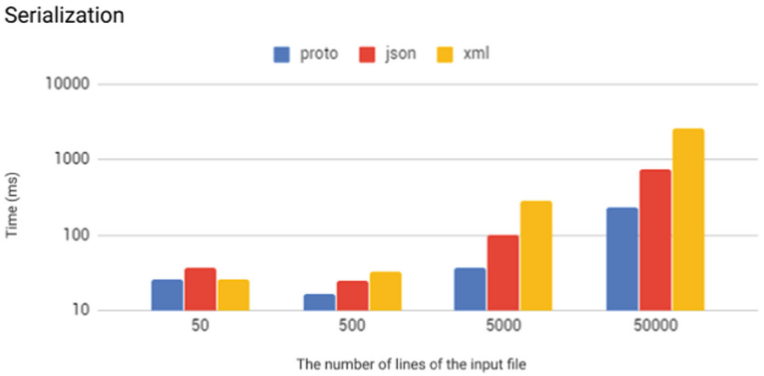

Fig. 7. Serialization time depending on the number of lines of the input file

As can be seen from the graph, in different tests the serialization time using Protobuf is the lowest compared to JSON and XML. XML serialization time is longer than JSON, except for the first test.

According to the results of four tests, Table 4 shows the generalized results of the deserialization time for the three methods. Figure 8 shows a graph of deserialization time depending on the number of lines of the input file.

Table 4. Time of deserialization of input data

The number of lines of the input file	Protobuf (ms)	JSON (ms)	XML (ms)
50	0,7	6,4	2,7
500	4,1	11,9	14,2
5000	41,1	91,4	111,6
50000	294,6	1071,4	1050,1

According to the simulation results, the following mathematical expression for the deserialization time using Protobuf was obtained:

$$t = \begin{cases} 0{,}014*n, & 0 \leq n < 50 \\ 0,7+7,556 * 10^{-3} * (n-50), & 50 \leq n < 500 \\ 4,1+8,222 * 10^{-3} * (n-500), & 500 \leq n < 5000 \\ 41,1+5,633 * 10^{-3} * (n-5000), & 5000 \leq n < 50000 \end{cases}$$

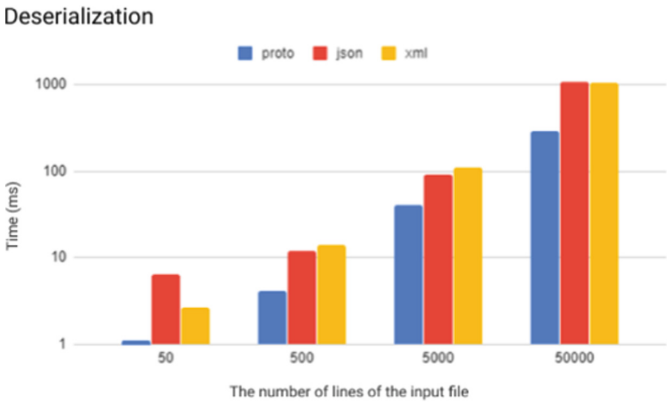

Fig. 8. Deserialization time depending on the number of lines of the input file

As can be seen from the graph, in the four trials, the deserialization time using Protobuf was the lowest compared to JSON and XML. In the first test, the XML deserialization time is much shorter than JSON, in other cases it is almost the same.

According to the results of four tests, Table 5 shows the generalized results of serialization and deserialization time for the three methods. Figure 9 shows a graph of serialization and deserialization time depending on the number of lines of the input file.

Table 5. Time of serialization and deserialization of input data

The number of lines of the input file	Protobuf (ms)	JSON (ms)	XML (ms)
50	27,1	44,3	28,5
500	21,3	37	48
5000	78,6	192,3	397,9
50000	526,2	1825,3	3645

According to the simulation results, the following mathematical expression for the total time of serialization and deserialization using Protobuf:

$$t = \begin{cases} 0{,}542*n,\ 0 \le n < 50 \\ 27,1 - 1,289 * 10^{-2}*(n - 50),\ 50 \le n < 500 \\ 21,3 + 1,273 * 10^{-2}*(n - 500),\ 500 \le n < 5000 \\ 78,6 + 9,947 * 10^{-3}*(n - 5000),\ 5000 \le n < 50000 \end{cases}$$

As can be seen from the Fig. 9, in different tests the total serialization and deserialization time using Protobuf is the lowest compared to JSON and XML. However, in the first test, Protobuf is almost indistinguishable from XML, although in other tests, as the number of lines of input increases, the difference increases.

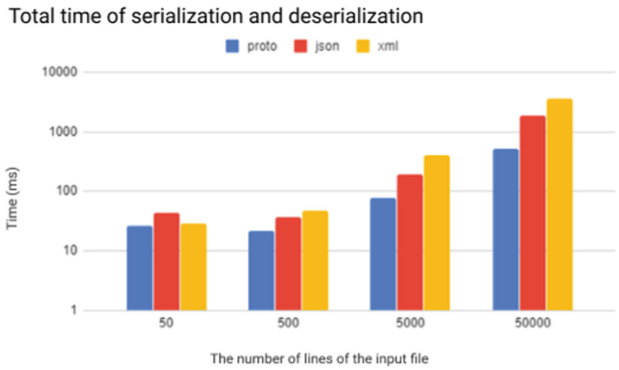

Fig. 9. Serialization and deserialization time depending on the number of lines of the input file

According to the results of four tests, Table 6 shows the generalized results of the sizes of serialized files for the three methods. Figure 10 shows a graph of the sizes of serialized files depending on the number of lines of the input file.

Table 6. Sizes of serialized files

The number of lines of the input file	Protobuf (kbit)	JSON (kbit)	XML (kbit)
50	41,6	52	71,4
500	402	501,6	688,8
5000	4005,5	4997,2	6862,1
50000	40040,7	49953,2	68606,3

According to the simulation results, the following mathematical expression is obtained for the size of the realized files using Protobuf:

$$s = \begin{cases} 0{,}832*n,\ 0 \leq n < 50 \\ 41,6 + 0,8*(n-50),\ 50 \leq n < 500 \\ 402 + 0,801*(n-500),\ 500 \leq n < 5000 \\ 4005{,}5 + 0{,}801*(n-5000),\ 5000 \leq n < 50000 \end{cases}$$

As can be seen from the graph, in all four tests the size of the serialized file using Protobuf is the smallest compared to JSON and XML. On the other hand, using XML, the largest serialized file sizes were obtained in all tests compared to Protobuf and JSON.

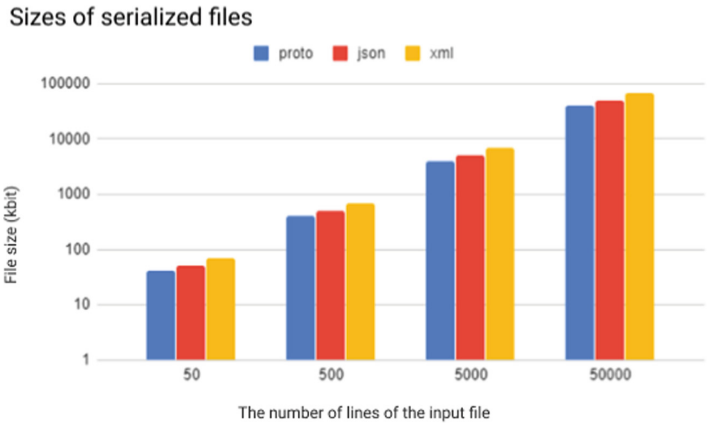

Fig. 10. The size of serialized files depending on the number of lines of the input file

As can be seen from the results of all four tests, the best results for all criteria were obtained using the Protobuf serialization method. It can be seen that as the volume of incoming data increases, the difference between serialization methods using XML, JSON, and Protobuf increases.

This is due to the fact that data with Protobuf is transmitted without metadata details, and this, in turn, does not add load to the payload. Also, Protobuf tightly compresses data, due to which the size of the serialized file is reduced in comparison with XML, JSON.

5 Conclusions

In this paper, the analysis of existing methods of information collection and processing in the IIoT network were analyzed, based on the prototype of the information collection and processing method was chosen.

The technique of collecting and processing informationhas been modified, which differs in the use of serialization based on the Protobuf method, which allows to increase

the efficiency of information processing over time and reduce the amount of transmitted information.

Full-scale modeling of the proposed solution due to the implementation of software based on a modified method. A feature of the method is the fixation of parameters- time and size of the serialized file. This simulation confirmed the efficiency of the proposed solution. Mathematical expressions were formed on the basis of the generalized results obtained in each test, which allow to determine the time and size of the serialized file for a given number of lines of the input file.

According to the simulation results, the proposed solution was evaluated - in comparison with JSON, Protobuf serialization time is 3.27 times less, deserialization time is 3.64 times less, total time is 3.5 times less, and file size is 1.25 times smaller.

References

1. Ven, R.: Three Industrial IoT Implementation Challenges (2018). https://dzone.com/articles/3-iiot-industrial-internet-of-things-implementatio
2. Khare, S., Totaro, M.: Big Data in IoT (2019). https://www.researchgate.net/publication/338365769_Big_Data_in_IoT
3. Khan, W.Z., Aalsalem, M.Y., Khan, M.K., Hossain, M.S., Atiquzzaman, M.: A reliable Internet of Things based architecture for oil and gas industry. In: 2017 19th International Conference on Advanced Communication Technology (ICACT), pp. 705–710. IEEE (2017). https://doi.org/10.23919/ICACT.2017.7890184
4. Lee, C.K.M., Zhang, S.Z., Ng, K.K.H.: Development of an industrial Internet of things suite for smart factory towards re-industrialization. Adv. Manuf. **5**(4), 335–343 (2017). https://doi.org/10.1007/s40436-017-0197-2
5. Ungurean, I., Gaitan, N.C., Gaitan, V.G.: A middleware based architecture for the industrial internet of things. KSII Trans. Internet Inf. Syst. **10**(7), 2874–2891 (2016). https://doi.org/10.3837/tiis.2016.07.001
6. Al Hadidi, M., Al-Azzeh, J.S., Tkalich, O.P., Odarchenko, R.S., Gnatyuk, S.O., Khokhlachova, Y.Y.: Zigbee, bluetooth and Wi-Fi complex wireless networks performance increasing. Int. J. Commun. Antenna Propag. **1**(48), 1–48 (2017). https://doi.org/10.15866/irecap.v7i1.10911
7. Shinde, Y.: Protobuf Performance Comparison (2016). https://dzone.com/articles/protobuf-performance-comparison-and-points-to-make
8. Babaria, U.: IoT Development Needs Microservices and Containerization (2018). https://www.einfochips.com/blog/why-iot-development-needs-microservices-and-containerization
9. Popenko, D., Kurdecha, V.: Analysis of object serialization methods for building a platform of big industrial data. Modern challenges in telecommunications, Kyiv, p. 222 (2020)
10. Demyd, P.: Protobuf application on industrialinternet of things. Prospects for development of information-telecommunication technologies and systems. Kyiv, p. 369 (2020)
11. Globa, L., Kurdecha, V., Ishchenko, I., Zakharchuk, A.: An approach to the Internet of Things system with nomadic units developing. In: 14th International Conference the Experience of Designing and Application of CAD Systems in Microelectronics (CADSM), Lviv, pp. 248–250 (2017). https://doi.org/10.1109/CADSM.2017.7916127
12. Globa, L., Kurdecha, V., Ishchenko, I., Zakharchuk, A., Kunieva, N.: The intellectual IoT-system for monitoring the base station quality of service. In: 2018 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom), Batumi, pp. 1–5 (2018). https://doi.org/10.1109/BlackSeaCom.2018.8433715

13. Globa, L.S., Skulish, M.A.: The model of multithread routing for broadband digital networks with integral service. In: 2008 18th International Crimean Conference - Microwave & Telecommunication Technology, Sevastopol, Crimea, pp. 376–377 (2008). https://doi.org/10.1109/CRMICO.2008.4676420
14. Senchenko, V.R., Koval, O.V., Globa, L.S., Novogrudska, R.L.: Intelligent modeling system based on cloud-technology. In: 2016 International Conference Radio Electronics & Info Communications (UkrMiCo), Kiev, pp. 1–4 (2016). https://doi.org/10.1109/UkrMiCo.2016.7739646