Edge computing-based, data-driven machine health and process monitoring system in the context of cyber manufacturing

VICTORIA GEISEL* and TIM PACK*, TH Köln, Germany

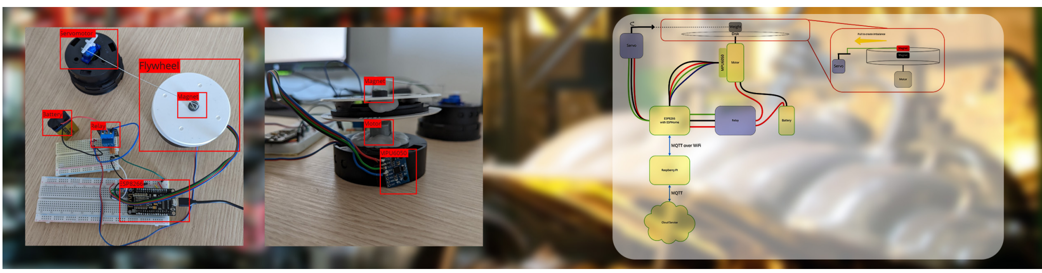


Fig. 1. schematic representation of the sensors and the project results

Theoretical and hardware prototype for data-driven machine health and process monitoring systems in the cyber-manufacturing sector based on the concept of edge computing.

Additional Key Words and Phrases: Edge Computing, Cloud Computing, machine health, cybermanufacturing, data-driven, process monitoring

ACM Reference Format:
Victoria Geisel and Tim Pack. 2022. Edge computing-based, data-driven machine health and process monitoring system in the context of cyber manufacturing. In *Mobile and Distributed Systems.* ACM, New York, NY, USA, 13 pages.

1 INTRODUCTION

Production machines in the factory sector are often working around the clock. To prevent a production downtime caused by a defective machine, the machine health, including the wear and tear of those, is analysed. Because of the geographically distributed machines within the manufacturing halls, a complex combination of software systems and hardware is necessary to stream and analyse large real-time datasets from the production machines. Therefore frameworks based on the concepts of fog- and cloud-computing are offered to aid the monitoring processes. An exemplary framework is introduced by Dazhong Wu et al. and described in the following:

*Both authors contributed equally to this research.

Mobile and Distributed Systems, WS21/22, TH Köln

1.1 Source literature

A technical paper from Dazhong Wu et al. titled "A fog computing-based framework for process monitoring and prognosis in cyber-manufacturing" served as a starting point of our researches [12]. The paper was published in 2017.
The paper refers to the area of cybermanufacturing as a modern manufacturing system in industry. The main focus of the paper is about monitoring the production machines (especially hydrostatic, coolant and hydraulic pumps) and making prognosis for them based on the collected data. The initial problem is the limited capability of existing systems to process the large volume of real-time data and to create predictive models concerning the machine health.
The paper results in a computational framework for fog-computing based cybermanufacturing systems. The proposed reference architectur is shown in the following illustration: As the illustration

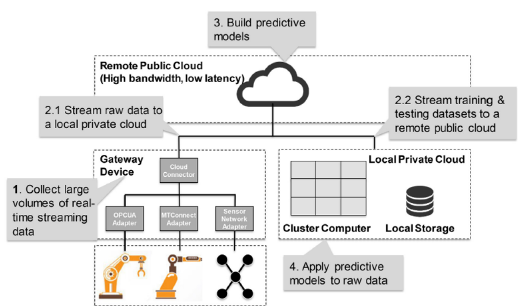


Fig. 2. Fog-computing based framework for monitoring and prognosticating [12]

shows, the framework consists of four steps. It starts with collecting the real-time streaming data from sensors and adapters applied to the production machines. The collected data ends in large data volumes, which are then streamed to a local private cloud in the edge and a remote public cloud. There are also training datasets for machine learning purposes streamed to the remote public cloud. In step three diagnostic and prognostic models are build based on the training datasets and machine learning algorithms. The models are then applied to the real-time streaming data in the local private cloud to do online prognosis and diagnosis on the health of the used machines [12].

1.2 Research Question

Based on the provided framework by Dazhong Wu et al. this paper concentrates on the application of this framework in a production factory to evaluate the necessity and the practicability of the framework. Therefore the paper answers the following research question: *How can an edge computing-based, data-driven machine health and process monitoring system in the context of cyber manufacturing be implemented in a factory?* In addition to the main research question the paper delivers an answer whether there are any restrictions in functionality for cloud- and fog-computing, which have an impact on the framework implementation. Moreover, this paper gives an insight on the performance time of use cases applied to the cloud and the fog.
In addition to the theoretical considerations, a practical simulator was created as a prototype and the following objectives for it were derived from the research question:

- Testing the network architecture and type of communication of the individual nodes in Edge-Computing
- To simulate an application of edge computing in the context of cybermanufacturing

- Analysis of the data generated by vibration of a motor with imbalance
- Comparison of the speed of classification of a data stream between edge device and cloud

2 RELATED WORK

Based on the contents of the source literature by Dazhong Wu et al. a literature research was done to collect further insights on cloud- and fog-computing and the cyber-manufacturing processes, to find similar frameworks to the one by Dazhong Wu et al. and to evaluate their framework concept. To find related papers several scientific libraries (e.g., Semantic Scholar, Connected Papers, Scopus and Google Scholar, ACM) where searched in. As research parameters the following key words and their combinations were used: Cyber-Manufacturing, Monitoring, Prognosis, cloud-computing, fog-computing, edge-computing, smart production machines, smart factory.
The result of our literature research was a list of about fifteen scientific papers. Three of them proved to be the most relevant and helpful concerning our project. Those three papers are summarized in the following:

2.1 “Internet of Things-based Fog and Cloud Computing Technology for Smart Traffic Monitoring”

The scientific paper *Internet of Things-based Fog and Cloud Computing Technology for Smart Traffic Monitoring* by Rajasekhara Babu et al. offers a fog- and cloud-computing framework in the application area of Traffic Monitoring [1]. Rajasekhara Babu et al. state the efficiency of basic cloud services in currently time sensitive application areas like oil, gas and traffic monitoring as "questionable". Therefore they propose an integrated architecture of fog- and cloud-computing, which should overcome the limitations of real-time analytics, latency and network congestion of those used basic cloud services. For Rajasekhara Babu et al. fog computing is a "complementary tool to improve the cloud performance" [[1], p. 15].
As queries in the application area of Traffic Monitoring they assume the count of vehicles, the calculation of traffic jam and the detection of incidents, which should be improved within their proposed architecture.
The paper includes a theoretical comparison of fog- and cloud-computing and a practical purpose, which contains a measurement of the response time for both implemented concepts. As the following illustration shows, the response time on the fog is significantly shorter than on the cloud.

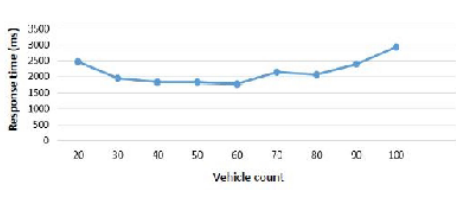


Fig. 3. Cloud-Computing response time [[1], p. 13]

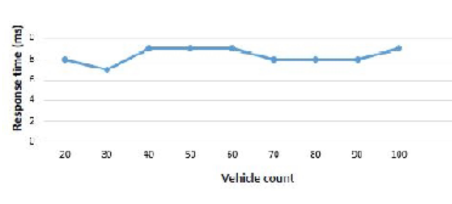


Fig. 4. Fog-Computing response time [[1], p. 14]

Their architecture, which integrates fog- and cloud-computing, served as the basic idea for our demonstrator architecture. The adjusted architecture can be found in section 3.2.

2.2 “Industrial Internet of Things Solution for Real-Time Monitoring of the Additive Manufacturing Process”

Salama et al. published a scientific paper titled *Industrial Internet of Things Solution for Real-Time Monitoring of the Additive Manufacturing Process* in 2019, which offers a real-time solution for

monitoring purposes in the additive manufacturing processes [9].
The architecture is based on the idea of the Industrial Internet of Things and resembles the architecture of Dazhong Wu et al. There is a physical layer, which contains two sensors, a Human-Machine-Interface Node, a Server Mirror Node and a Control Unit. The both used sensors, a temperature sensor and a filament breakage detection sensor, are attributed to the application area of Additive Manufacturing. The Communication Layer manages the information exchange between the node network and the application network by collecting, splitting and sending the information from or to other nodes. The third architecture layer is the Application Layer, which monitors and manages the whole process of Additive Manufacturing. As the description of the architecture reflects the proposed concept does not intend to use the concepts of fog- and cloud-computing. But the paper is of higher relevance for our project because of the similarities in the proposed architectures [9].

2.3 "Machine vision based condition monitoring and fault diagnosis of machine tools using information from machined surface texture"

The third relevant paper is *Machine vision based condition monitoring and fault diagnosis of machine tools using information from machined surface texture* from Yuekai Liu et al. published in 2021 [4]. The paper reviews tools and techniques for condition-based maintenance in the metal production, respectively the metal removal manufacturing and the metal additive fabrication. The tools and techniques concentrate on a machine vision based condition monitoring and fault diagnosis of machines, which aren't controlled by human-beings. Because of those unmanned automation production the machines are equipped with many sensors measuring the machine health.
To process the large volume of collected sensor data the edge computing concept is presented to compute the data locally. The following illustration represents the framework of fog-computing based cyber manufacturing systems: The presented framework consists of three layers: The *Terminal*

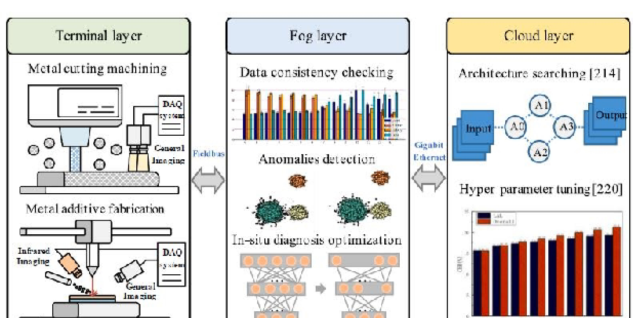


Fig. 5. Fog-computing based framework in cyber manufacturing systems [[4], p. 22]

Layer, *Fog Layer* and *Cloud Layer*. The sensors as part of the Terminal Layer are applied to the machines and collect data continuously. The data is then uploaded to the Fog Layer, where the consistency of the data gets checked, data anomalies get detected and diagnosis are made on the machine. Therefore the Fog Layer preprocesses the data, which leads to the usage of less computing resources, which would be needed if the data was processed on the Cloud Layer. The Cloud Layer receives the preprocessed data and saves it. Moreover, the Cloud Layer processes tasks, for which high computational resources are needed [4].

3 THEORETICAL PROTOTYPE

3.1 Onsite vs Edge vs Cloud

Edge Computing is listed as one of the emerging technologies in the Gartner Hype Cycle of 2021 [2]. The Hype of Edge Computing can be explained because of the given restrictions and limitations of Cloud Computing. The usage of cloud solutions has, among other aspects, experienced greater disadvantages concerning a real-time process controlling as well as the streaming of large data volumes into the cloud. Edge Computing aims to eliminate those disadvantages by processing the data on the edge of the networks. In particular, the concept of Edge Computing can solve the problems of delayed communication within monitoring processes in the industry [[4], p. 22]. The differences between Cloud- and Edge Computing are shown in the following table (on the basis on [1]).

Attribute	Cloud Computing	Edge Computing
distribution	centralized	distributed
latency	high	low
distributed analytics	analytics only at cloud	analytics distributed between cloud and edge nodes
data analytics	data aggregation at cloud; decisions from cloud	data aggregation partially at edge nodes and partially at cloud; quick decisions from edge node

Following the concept of Edge and Cloud Computing, the artifacts of the found literature sources were used to create an own prototype, which will be described in the following, and which is illustrated in figure 6.

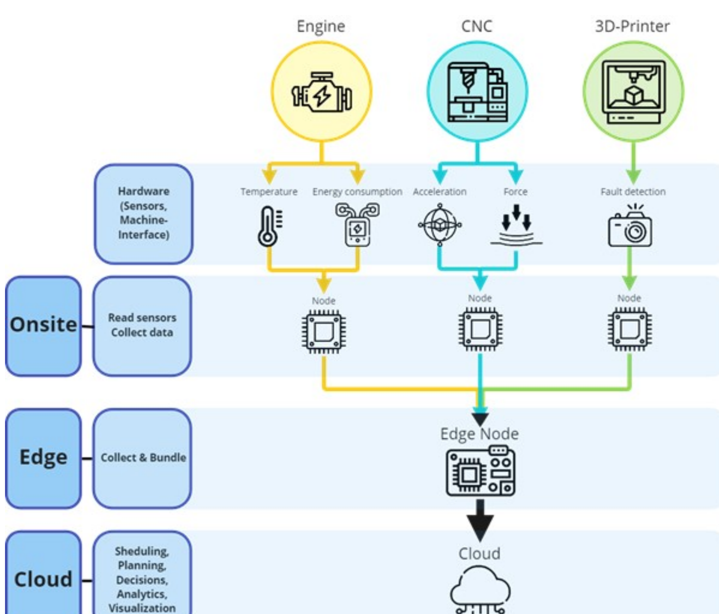


Fig. 6. Illustration of the physical layout of the components and their specific tasks of a system which, supported by edge computing, can check and monitor the condition of the machines in a factory hall.

The prototype is designed to monitor pumps, similar to the prototype of Dazhong Wu et al., and CNC-Machines (Computerized Numerical Control Machines). As an addition the idea of Salama et al. was absorbed to check on the health of 3D-printers during additive manufacturing processes (the implementation is not presented within this paper, but is part of future work (see section *Outlook*)). The prototype is a stand-alone and is not embedded in any specific manufacturing factory.
The mentioned machines are, in addition to their machine interface, equipped with sensors to collect machine-specialized data. Sensors to measure the temperature and the energy consumption are added to the pump. The CNC machine is provided with an acceleration and force sensor.
To get data of high quality the position of the sensors is important. Whereas the acceleration sensor is positioned at the worked-on workpiece, the dynanometer to measure the force (pressure) is installed at the table, on which the CNC machine is standing. The both sensors, temperature and energy consumption, are positioned on the motor of the pump.
Applying the concepts of Edge- and Cloud-Computing, the aim for the monitoring process on each level has to be defined. Onsite the sensors are read and collecting data. On the edge the data gets collected and bundled and first quick data analytics processes are performed. The cloud schedules and plans the processes related to the manufacturing process (e.g., planning maintenance processes based on the collected and analysed sensor data). Moreover, in the cloud decisions are made, data analytics are processed and their results are visualized.
The following section describes the specific use cases of our prototype.

3.2 Use Case

The basic idea of our prototype is to read the sensor data, processing the data on the edge and on the cloud and comapre the measured response times to make a statement on the performance of cloud and edge.

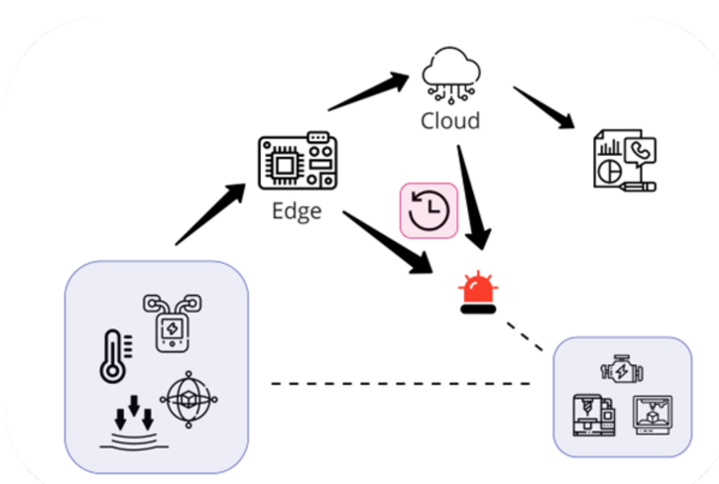


Fig. 7. General concept of a possible implementation of a prototype to test the applicability of the theoretical background of cybermanufacturing. [own illustration based on [1]]

Therefore, as figure 7 shows, the sensor data is transmitted to the edge and forwarded by the edge to the cloud. Both, cloud and edge, process the same use cases, for which the execution time gets measured. In addition to that, the cloud does data analytics and, as a result, offers reports concerning the machine health.

To be able to compare the measured times, the use cases are executed with the same conditions and parameters. The use cases are, as shown in figure 8:

- **temperature transgression:** The temperature sensor gathers the temperature of the motor of the pump.
- **voltage transgression:** The energy sensor assembles the energy consumption of the motor.
- **imbalances:** The accelearation sensor measures vibrations and off-beat movements.
- **force transgression:** The dynanometer collects data of the current pressure of the milling head

For every incident (exceeding and falling below fixed metrics) an alert is sent and the machine gets stopped. The aim is to reduce the risk of aggravating the machine health by not stopping the machine directly after the inconsistentcy is detected.

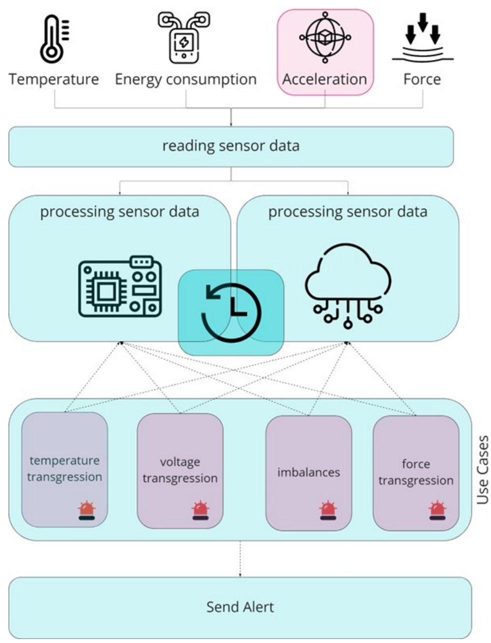


Fig. 8. Concept of measuring use cases for splitting the data sensor from machines in factories between edge and cloud. [own illustration]

4 HARDWARE PROTOTYPE

Part of the research on edge computing in the context of cybermanufacturing was the development of a hardware prototype that would help to better understand the work with the sensors and to implement the architecture of such a system in a real world application. The main focus here was on the communication of the individual network components and the simulation of the data described in the underlying paper [12], when measuring the vibration of an electric motor to determine the machine health of the motor. In the following, the architecture and development of the simulator is described in more detail and the procedure is explained. The source code of the project is publicly available here: GitHub - edgecybermanufacturing

4.1 Architecture

The simulator planned and implemented in this project can be roughly classified into three main components. The *hardware part*, *the MQTT broker* and *the Python scripts* for evaluating the data. The hardware reads the data from the sensor, controls the motor and forwards the data to the MQTT broker. From there, the data is forwarded to a Python script that evaluates the data and can send commands to the hardware to stop the motor, for example. It was planned that this evaluation would be carried out on a device typical for the application of edge computing (here a Raspberry Pi 4) and on a cloud server. Afterwards, the latencies of the two systems were to be compared with each other.
The hardware consisted of a NodeMCU (Figure 10) connected to an MPU6050 sensor and a controllable relay. A 9V battery and a motor was connected to the relay. The motor drove a (Figure 10) flywheel, on which a magnet was attached. The motor was mounted in a round housing, which was glued to a tabletop. The sensor for measuring the vibration and oscillations caused by the rotation was located directly on the motor housing. In this way, the motor could be controlled (on/off) via the NodeMCU and the data from the MPU6050 could be read out at the same time. Furthermore, a laptop running the MQTT Broker and a Raspberry PI 4 as the edge computing device running an evaluation script were used. A dedicated server was used for the cloud (1 vCore, 2GB Ram, 40GB SSD NVMe, 250 Mbit/s network). ESPHome, a custom firmware, was installed on the NodeMCU. This took over the control of the relay and the reading of the MPU6050 sensor. In addition, a simple forwarding rule to the MQTT server could be configured in ESPHome. This allowed the data to be read and analysed directly from a topic of the MQTT broker. A Python notebook was created for the analysis. The Python scripts running on the Raspberry PI and the cloud server for evaluation were connected to the MQTT server using the library (Phao MQTT). The MQTT broker ran on a local laptop, which forwarded the sensor data for the cloud server to an MQTT broker accessible in the public network.

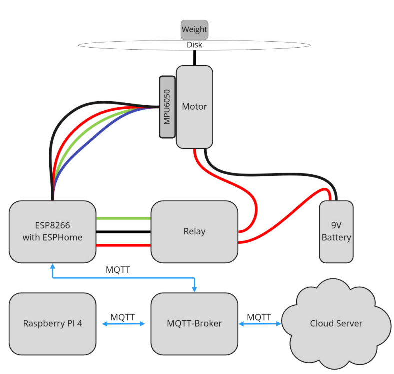


Fig. 9. Schematic representation of the simulator's structure. Above, the engine with the flywheel and the sensor. In the middle, the ESP8266, the relay and the battery. At the bottom, the Rapsberry PI 4 (edge device), the local MQTT broker and the cloud server.

4.2 Implementation and experimental design

First, the data was recorded with the simulator. For this purpose, a 2.5gr weight was attached to the flywheel. The motor was connected to a 9V battery and the values of the accelerometer were recorded for a time span of 5 minutes. This recording was then repeated three times, moving the weight outwards by 0cm, 2.5 cm and 3.5 cm in order to generate a different degree of imbalance and to be able to later compare the values of the sensor attached to the motor casing. The data generated by this experiment was then analysed and compared. Afterwards, it was to be classified with the help of a machine learning model. However, it was later discovered that this comparison did not yield any results because the classification model trained with the data did not provide reliable results. In order to compare the classifications on the different devices anyway, an already existing model with a comparable data basis was used. For this purpose, the "Condition monitoring of hydraulic systems Data Set" [7] was used and a pre-configured model [6] was trained according to specifications. This trained model was then embedded in a Python script which, as soon as it receives a message in a certain topic, classifies a data set and, as soon as it is finished, writes a response message in the MQTT topic. This script ran on a cloud server and on an edge device at the same time. Another script sent the initial messages to the MQTT topic and recorded the response times for the two test clients for later evaluation. This was done twice. Once without performing the classification on the clients and once with. There were 100 runs per test during which the times were recorded.

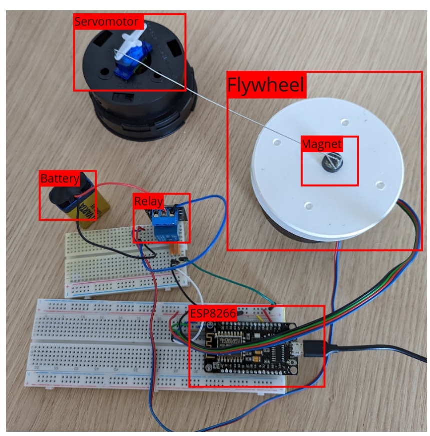


Fig. 10. A photo of the simulator with all hardware components. The servo motor shown here was not used in the later tests because the friction was too high and prevented rotation.

5 RESULTS

The findings of the measurements and analyses that were performed throughout the course of this project are briefly described below. The outcomes of the simulation-based measurements are described first. The measurement findings from the evaluation of edge and cloud are then described.

5.1 Simulator

Figure 12 shows the sensor data of the X-axis of the accelerometer in metres per second squared (m/s²) during the test with the simulator. The simulator was operated for 5 minutes with a 9 volt battery. This was done 3 times, with a magnet (2.5gr) stuck further out (0cm,2.5cm,3.5cm) on each run to change the imbalance. The data of the run with the weight at 0cm ranged from 2m/s² to 3m/s², at 2.5cm between -2.1m/s² and 8.4m/s² and at 3.5cm between -2.3m/s² and 8.2m/s². The recording at 3.5cm stops at measurement 75 and does not provide any more data because during the test the violent movements and friction caused by the high unbalance broke and melted the inside of the electric motor and the housing. A Fast Fourier transformation was also performed (see Figure 13). This shows the frequency components of the individual signals. Here a peak to -25 at 3.5cm can be seen at run 19.

5.2 Comparison

The data for the comparison between the cloud server and the edge device is shown in Figure 11. The mean times of the two devices for all tasks are shown here. The mean ping time for the edge device is 253ms and for the cloud 84ms. For the classification tests, the mean time for the edge device was 243ms and for the cloud 102ms. The standard deviation for all tests was between 60ms and 70ms.

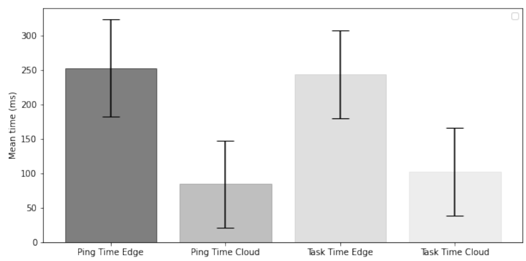


Fig. 11. The average times for 100 runs (N=100) during the comparison between Cloud and Edge in milliseconds with the test classification and without.

6 LIMITATIONS AND DISCUSSION

6.1 Limitations

The data recorded with the simulator should be viewed critically, as it was recorded via the MQTT broker with a frequency that is too low for this type of sensor. This is around 30-70 Hz and can strongly influence the results of the filters applied to the data to analyse it [8]. In addition, it should be noted that the strength of the motor's vibration also depends strongly on the mounting, the current strength, construction and many other factors that could not be considered further here due to time constraints [5]. Another factor to be noted is the direction of the sensor on the motor housing. If the data is analysed and the orientation of the sensor changes after the analysis, the data could be very different from the previously recorded data.
In the data for the comparison between edge and cloud computing, it should be noted that the tests were carried out in a controlled environment in a home network. Any deviations that could occur in a network on a factory floor were not taken into account. In addition, the comparison only analysed data from a specific sensor and compared the speed of the analysis. On a factory floor, a much higher amount of data can be expected.

6.2 Discussion

The data from the simulator shows clear differences between the runs without unbalance and a little unbalance, but the different levels of unbalance are difficult to separate. A deeper analysis is needed to be able to make more precise statements [10]. Based on the results, it can be said that it is possible to detect differences in the extent of vibration by machine, but this could not be tested because the data could not be sufficiently analysed. The signal can also be analysed by the fourier transformation [3], but this requires a very precise determination of the recording frequency of the data, which was not given in this experiment. When comparing the cloud server and the edge device, it was to be expected that the edge device would have a significantly lower average ping value than the cloud server, as it is located in the same network as the MQTT broker and the script that records the times. However, the results show a clear shift in all values between Edge and Cloud, both for the average and the individual values in Figure 14. This could be related to the fact that the slower hardware of the edge device is already busy with the MQTT process. For this reason, the values from the comparison with the classification are difficult to compare, as the same factor could play a role here and the difference is not due to the time required for the classification, but to the workload of the MQTT client on the system.

7 CONCLUSION

In the theoretical exploration of the topic of edge computing in the context of cybermanufacturing, different possibilities of monitoring machines using sensors were analysed and separated into edge and cloud with regard to the type of data processing. In this way, the approach of Dazhong Wu et al. was taken up, its feasibility tested and the theoretical model extended.
The theoretical prototype was designed based on several scientific publications. The results of the literature research has shown, that the concepts of Edge- and Cloud Computing are applied in a very similar way, not depending on the application area it is used in. In comparison to other papers, especially [12] and [1], the theoretical prototype assumes that the production machines are unmanned (see [4]) and there is no human being, receiving the alerts to initiate further actions on the machine.
In addition, different application examples were shown and a special case was tested using a simulator as a model example. During which it became clear that from a hardware point of view, such an architecture can be easily implemented with a central MQTT broker. It was easy to communicate the data from the edge to the cloud, as the MQTT broker used (EMQX) can forward the data to other brokers or cloud services via a bridge function and by configuring various rules. The simulator has also shown that the difficulty lies more in the correct use and interpretation of the data than in the technical implementation of the hardware and communication. In addition, the electric motor used in the simulator enabled a typical cybermanufacturing application to be simulated, although it should be noted that the pumps and motors used in this context are much more robustly built and also behave significantly differently in the event of wear. The analysis of the data was partially carried out, but since further analysis would have required deeper knowledge in the field of signal processing, it was not pursued further. Finally, a comparison was also made between cloud and edge devices. Although the data obtained from this did not allow any precise statements to be made, further tests can easily be implemented with the resulting code in order to find out, for example, from which amount of data the cloud is faster than the edge device.

8 OUTLOOK

With the simulator developed during this project, further data could be recorded and the vibrations analysed in more detail. In addition further sensors such as a current measurement or the temperature already built into the IMU can be used to improve the fault and condition prediction and make it more accurate. The analysis of the simulator data could be further improved by first defining an uniform recording frequency of the sensor data that is not limited by the MQTT broker. Comparable experiments [10], for example, use "order tracking" to convert the time sequence of the vibration into a rotation sequence, which can subsequently be analysed using the vibration frequency. In addition, the microcontroller that reads the data from the sensor could collect the data with a predefined frequency and then send it as data packets to the MQTT broker. This would ensure a stable frequency, which is very important for a reliable evaluation. [8] The original idea of the simulator was to make the strength of the unbalance adjustable by a servo motor attached to the weight, and thus to make it an adjustable parameter in the experiment. However, this turned out to be too complex during assembly, which meant that it could not be implemented. In the future this could make it possible to increase the unbalance during the comparison of the two systems and to test which system would send the command to switch off the motor first. Another question that could be explored in future research would be at what number of sensors for a simple analysis, such as the classification of data from a single sensor in this paper, the calculations become too complex to be performed on an edge device. Furthermore, the question could be investigated up to which degree of hardware equipment an edge device is worthwhile via the use of a cloud server, or whether it only depends on the respective use case. [11] During the research on this project, it has become clearer which tasks are more suitable for pre-processing on an edge device than others. In the future, this separation could be analysed in more detail and a general model could be developed to determine how to partition which data in the context of cyber manufacturing.

REFERENCES

[1] Swati Dhingra, Rajasekhara Babu Madda, Rizwan Patan, Pengcheng Jiao, Kaveh Barri, and Amir H. Alavi. 2021. Internet of things-based fog and cloud computing technology for smart traffic monitoring. *Internet of Things* 14 (2021), 100175. https://doi.org/10.1016/j.iot.2020.100175

[2] Gartner. 2022. 5 Trends Drive the Gartner Hype Cycle for Emerging Technologies, 2020. https://www.gartner.com/smarterwithgartner/5-trends-drive-the-gartner-hype-cycle-for-emerging-technologies-2020

[3] Sangbo Han. 2010. Measuring displacement signal with an accelerometer. *Journal of Mechanical Science and Technology* 24, 6 (2010), 1329–1335. https://doi.org/10.1007/s12206-010-0336-1

[4] Yuekai Liu, Liang Guo, Hongli Gao, Zhichao You, Yunguang Ye, and Bin Zhang. 2022. Machine vision based condition monitoring and fault diagnosis of machine tools using information from machined surface texture: A review. *Mechanical Systems and Signal Processing* 164 (2022), 108068. https://doi.org/10.1016/j.ymssp.2021.108068

[5] Raj Mistry, Bill Finley, Scott Kreitzer, and Ryan Queen. 9/8/2014 - 9/10/2014. Influencing factors on motor vibration & rotor critical speed in design, test and field applications. In *Petroleum and Chemical Industry Technical Conference (PCIC), 2014 IEEE*. IEEE, JAPAN,GERMANY, 227–236. https://doi.org/10.1109/PCICon.2014.6961887

[6] mjain72. 25.08.2022. Condition monitoring of hydraulic systems using xgboost modeling. https://github.com/mjain72/Condition-monitoring-of-hydraulic-systems-using-xgboost-modeling

[7] Nikolai Helwig, Eliseo Pignanelli, Andreas Schätze. 25.08.2022. Condition monitoring of hydraulic systems Data Set: The data set addresses the condition assessment of a hydraulic test rig based on multi sensor data. Four fault types are superimposed with several severity grades impeding selective quantification. https://archive.ics.uci.edu/ml/datasets/Condition+monitoring+of+hydraulic+systems

[8] Andrii Rudyk, Viktoriia Rudyk, Nazarii Radul, and Mykhailo Matei. 2019. The influence of the program code structure on the discretization frequency of the onboard computer of a mobile robot. *Modeling, Control and Information Technologies: Proceedings of International scientific and practical conference* 1, 3 (2019), 212–214. https://doi.org/10.31713/MCIT.2019.71

[9] Mahmoud Salama, Ahmed Elkaseer, Mohamed Saied, Hazem Ali, and Steffen Scholz. 2019. Industrial Internet of Things Solution for Real-Time Monitoring of the Additive Manufacturing Process. In *Information systems architecture and technology*, Leszek Borzemski, Jerzy Świątek, and Zofia Wilimowska (Eds.). Advances in Intelligent Systems and Computing, Vol. 852. Springer, Cham, 355–365. https://doi.org/10.1007/978-3-319-99981-4{_}33

[10] therandomwalk. 2021. Motor Vibration Analysis - The Random Walk. https://therandomwalk.org/wp/motor-vibration-analysis/

[11] Blesson Varghese, Nan Wang, Sakil Barbhuiya, Peter Kilpatrick, and Dimitrios S Nikolopoulos. 2016. Challenges and opportunities in edge computing. In *2016 IEEE International Conference on Smart Cloud (SmartCloud)*. IEEE, 20–26.

[12] Dazhong Wu, Shaopeng Liu, Li Zhang, Janis Terpenny, Robert X. Gao, Thomas Kurfess, and Judith A. Guzzo. 2017. A fog computing-based framework for process monitoring and prognosis in cyber-manufacturing. *Journal of Manufacturing Systems* 43 (2017), 25–34. https://doi.org/10.1016/j.jmsy.2017.02.011

A APPENDIX

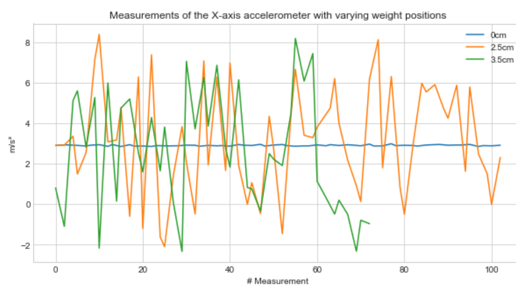


Fig. 12. The times measured in the test without a task and only the delay in transmission (ping time)

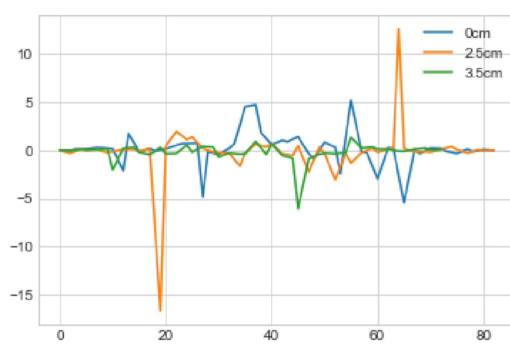
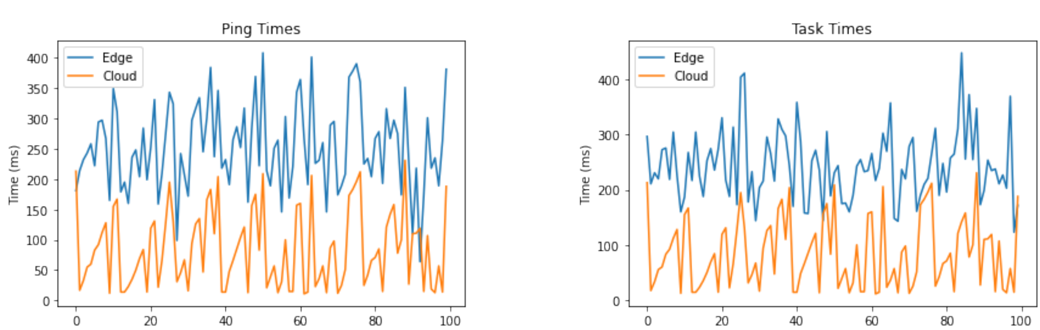


Fig. 13. A Fast Fourier Transform with the data of the measurements from the simulator on the X-axis.



(a) The times measured in the test without a task and only the delay in transmission (ping time)

(b) The measured times in the test with a classification task (task time).

Fig. 14. The measurements of the ping times (without classification) and task times (with classification) of the cloud server and edge device.