FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

FH Aachen

Faculty
Electrical engineering and information technology

Bachelor Thesis

Design and Implementation of a Performance Measurement System for an Industrial Sewing Machine

Nicolas Harrje
Matr.-Nr.: 3518047

Referent: **Prof. Dr-Ing. ...**

Korreferent: **Prof. Dr.-Ing. ...**

August 15, 2025

Erklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe.

Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht.

Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen.

Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Aachen, August 15, 2025

Geheimhaltung - Sperrvermerk

Die vorliegende Arbeit unterliegt bis [Datum] der Geheimhaltung. Sie darf vorher weder vollständig noch auszugsweise ohne schriftliche Zustimmung des Autors, des betreuenden Referenten bzw. der Firma [Firmenname und -sitz] vervielfältigt, veröffentlicht oder Dritten zugänglich gemacht werden.

Contents

1. Introduction

Industrial sewing machines are of crucial importance in the textile industry, where they are typically utilized in the final stage of production to assemble the end product. This stage necessitates the highest level of human involvement, thereby becoming a pivotal element in determining both production efficiency and product quality. Therefore, the implementation of performance measurement techniques is particularly appropriate in this context. In the field of performance measurement, the seminal work by Neely [Neely et al., 1995] is widely cited. They defined performance measurement as ”[...] the process of quantifying the efficiency and effectiveness of action.” . This is frequently achieved through the implementation of Key Performance Indicators (KPIs), as they are formally standardized in the ISO 22400 framework, which governs operations management and production.

In recent years, the popularity of automatic systems for performance measurement on sewing machines has increased. Nonetheless, these systems continue to encounter certain challenges that have frequently been overlooked. Firstly, it must be acknowledged that a considerable number of systems are dependent on cloud technology. This reliance engenders certain issues, including elevated latency and the perpetual financial obligations associated with cloud usage. Secondly, the utilization of standards and frameworks is frequently neglected, which results in the complexity of scaling and maintaining these systems. Thirdly, the prevailing focus of numerous works in this field is retrofitting sewing machines, rather than utilizing the machine’s inherent data, which often leads to the production of erroneous results. Fourthly, the dearth of software architecture that utilizes services engenders considerable challenges in achieving scalability.

The objective of this thesis is to establish a replicable methodology for designing and implementing a performance measurement system, with a sewing machine serving as a case study. This encompasses the provision of an overview of standards frameworks and technologies, in addition to the demonstration of the selection process for the most suitable option and its subsequent implementation. This thesis proposes a system that maximizes the use of actively maintained open-source technologies while ensuring ease of scalability for future expansion. The end result will be a dashboard that provides the most important KPIs (such as cycle time, OEE, setup time, and down time) in real time.

The scope of this work is limited to a Brother sewing machine of type UF-8910, which is connected to a WAGO PLC of type 750-8101 PFC100 CS 2ETH. The WAGO PLC employs the OPC-UA protocol for data transmission to the network. The anticipated data flow and signals are modeled and do not originate from the physical sewing machine and PLC. The sewing machine is part of a shop-floor that is used for workshops where industry customers can gain insight into productivity and quality enhancements within production environments through digitization. The system outlined in this thesis is intended to serve as a demonstrative model, and as such, it will feature visualizations that elucidate its real-time capabilities. The derivation of KPIs must be constrained to those that require querying from the database without necessitating additional post-processing.

The relevance of this thesis is predicated on the increasing demand for data-driven decision-making to optimize efficiency and reduce costs, a phenomenon that is especially pronounced in the highly competitive garment industry. Performance measurement systems empower production management to identify inefficiencies and minimize unproductive periods. Furthermore, they furnish actionable insights that facilitate targeted operator coaching. This thesis makes a significant contribution to the extant knowledge base concerning IoT-based monitoring systems, as it focuses on replicable methods. This as well as the focus on open source technology make the system architecture well suited for small and medium sized companies with limited resources.

This thesis employs a design-science approach, with the sewing machine performance measurement system serving as the artifact. A literature review is also employed to provide a comprehensive overview of the extant related work, as well as the frameworks, standards, and technologies relevant to the subject. To further implement suitable technology solutions, a structured technology selection process is being developed and followed.

In the following, the structure of this thesis is being outlined. The initial section presents the foundational technologies, frameworks, standards, and other groundwork upon which this work is based. The related work section then reviews relevant literature, examining systems with similar objectives to contextualize and position the approach proposed in this thesis. Subsequently, the requirements and system design section details the specific needs addressed by the system, as well as the selected KPIs and technologies. Building on this foundation, the implementation of the system is described. The subsequent evaluation section provides an analysis of the system's strengths and limitations. Finally, the outlook and conclusion offer a summary of the findings and discuss potential directions for future development.

2. Foundations

2.1. Setting

The sewing machine in question is a Brother UF-8910.It is part of a textile production line in which a wristband is being produced. The wristband contains an RFID chip capable of storing diverse information. For instance, the technology could be utilized to store a personnel identification number. In the event that the technology is scanned at a workplace, the workplace's dimensions would adapt to conform to the worker's size. The production line has not been designed for implementation in actual industrial-scale manufacturing; rather, it functions as a model environment for workshop purposes. The sewing machine is utilized during the final stage of the wristband assembly process. The utilization of the machine is characterized by a straightforward procedural nature. The objective is to sew both ends of the open wristband together with a single seam to close it. The following figure illustrates this procedure.
figure

The signals that can be extracted from the sewing machine encompass:

- Second walking foot stroke: Two walking foot strokes can be set for different thikness of materials. This signal indicates if the walking foot is currently in the second stroke hight.
- Thread trimming: Indicates when the thread is trimmed
- Pressure foot: Indicates if the pressure foot is lifted
- Upper shaft rotating: Indicates if the machine is actively sewing
- Other than home screen: Indicates that the menu screen is currently not in the home screen
- Home screen and not sewing: Indicates that the menu screen is in the home screen and the machine is not actively sewing

The subsequent figure illustrates the various components of the sewing machine, thereby facilitating a more profound comprehension of the signals.
figure

2.2. Definitions

2.2.1. Takt Time

Maximum time allowed to produce one product in order to meet customer demand

2.2.2. IoT and IIoT

The term Internet of Things was first coined by [Ashton,] when explaining the idea of combining RFID with the internet in an executive meeting. He explains that on the "normal" internet, most

of the content is created by human beings. In contrast to this in the Internet of Things the data is generated by things and often describes things. But his emphasizes lays more on the description of things. For example to track and count them. The information to do so would come from sensors and RFID, he says. Of course in these days more of the information on the internet is generated by bots and AI. But other than that the distinction still holds true.
The Internet Society [Rose et al.,] further explains that in the Internet of Things, machines are communicating with each other and are addressable via an own IP address. This standardizes the way in which devices communicate. They also mention that "Today, the Internet of Things has become a popular term for describing scenarios in which Internet connectivity and computing capability extend to a variety of objects, devices, sensors, and everyday items."
The Industrial Internet of Things is just the description of a domain where the IoT is used. In this case in manufacturing. [Wha,]

2.3. State of the Art

2.3.1. Industrial IoT Architectures and Patterns

Due to the requirement that the solution be developed utilizing IoT technologies and is set within a production context, a review of Industrial IoT (IIoT) architectures and patterns was conducted. The Industrial Internet Reference Architecture (IIRA) [Young, 2022] serves as a comprehensive framework, offering valuable insights into various architectural models and design patterns relevant to this domain. This reference architecture describes the following patterns: IoT Component Capability Pattern, Three-Tier Architecture Pattern, Gateway-Mediated Edge Connectivity and Management architecture pattern, Digital Twin Core as a Middleware Architecture Pattern, Layered Databus Architecture Pattern, System-of-Systems Orchestrator Architecture Pattern. Of these patterns only the first two are applicable within the scope of this work. Therefore the other ones will only be described on the surface.

Architecture Patterns IoT architecture patterns define the structure and operation of various IoT systems, detailing their implementation and highlighting their unique characteristics.

IoT Component Capability Model Pattern A single component and its associated capabilities are described, with the possibility that a component may comprise multiple sub-components. Consequently, the entire system can also be regarded as a component. The specific meanings of the capabilities are illustrated in the accompanying figure.

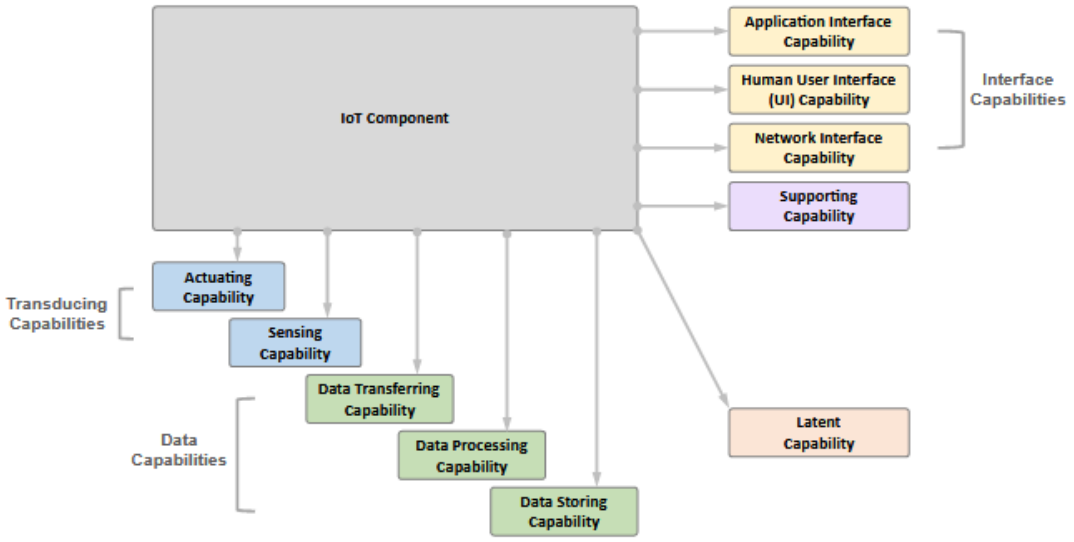

Figure 2.1.: Component Capability Pattern.
(Young et al., 2022, S. 40)

Three-Tier Architecture Pattern The system comprises the Edge, Platform, and Enterprise Tiers, as well as connecting networks. The Edge Tier contains sensors and gateways that collect data. These are connected by the Proximity Network. Data preprocessing may already be happening there.
The Platform Tier is responsible for most data processing and storage via databases. It is connected to the Edge Tier via the Access Network.
The Enterprise Tier provides domain-specific applications and interfaces for end users. These are built upon the processed data from the platform tier. It also issues controls to lower tiers. This tier is connected to the Access Network via the Service Network. The three tiers can also be further divided into different domains. That makes sense for bigger systems. But for a simple system as the one described in this work it is not necessary and therefore these domains will not be explained here.

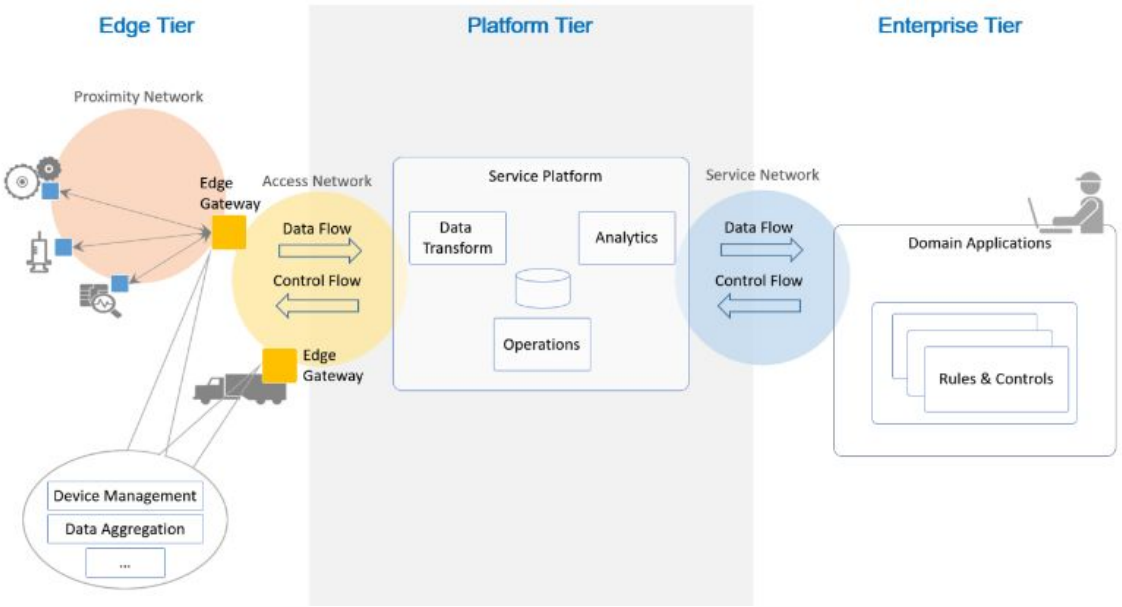

Figure 2.2.: Three Tier Architecture
(Young et al., 2022, S. 44)

2.3.2. KPIs and Metrics for Performance Evaluation in Sewing Operations

Performance measurement in the textile production industry is important due to intense competition. It enables producers to identify potential bottlenecks, provides a deeper understanding of processes, and facilitates more effective resource allocation [Alauddin, 2018].
Key performance indicators (KPIs) and metrics serve as essential tools for performance measurement. Kang et al. [Kang et al., 2016] from the National Institute of Standards and Technology in the United States analyzed the relationships among various types of KPIs and metrics used in operations management and production, based on the ISO 22400 standard. Supporting elements, referred to as metrics in this thesis, describe the measured data necessary for calculating basic KPIs. These supporting elements are categorized into time and quantity. Time elements quantify the duration of various events, such as the production time per unit. Conversely, quantity elements pertain to the number of produced items. Maintenance elements capture information about machine-related issues.
Based on the supporting elements, basic KPIs can be calculated. These KPIs are categorized into production, quality, and maintenance KPIs.
The researchers also emphasize the importance of comprehensive KPIs, which provide a broader overview of production performance. These KPIs build upon basic KPIs and include, for example, Overall Equipment Effectiveness (OEE), which is calculated by multiplying the KPIs for availability, performance, and quality ratio.
Other studies [Kiron, 2022, Alauddin, 2018] have specified KPIs specifically for the sewing section of a textile production plant. In this context, some KPIs overlap with those examined by Kang et al., while additional KPIs unique to the sewing section have also been introduced. The following table classifies these sewing-specific KPIs within the hierarchical framework proposed by Kang et al.

Table 2.1.: Classification of KPIs and Metrics in Sewing Section (based on Kang et al., 2016 and ISO 22400)

KPI/Metric	**Description**	**Classification (Kang et al., 2016 / ISO 22400)**
Supporting Element: Time		
Cycle Time	Total time taken to complete one operation, from start to start of the next piece.	Supporting Element: Time
Standard Minute Value (SMV)	Time required to complete a specific job under standard conditions and pace.	Supporting Element: Time
Allowance	Extra time permitted for personal needs, delays, and fatigue in production.	Supporting Element: Time
Idle Time/Machine	Time when operators or machines are not working, considered lost time.	Supporting Element: Time
Supporting Element: Quantity		
Operation	A step in the process required to convert materials into a finished product.	Supporting Element: Quantity
Manpower to Machine Ratio	Ratio of workers to machines, used to optimize labor and production.	Supporting Element: Quantity
Absenteeism	Rate of operator absence, which affects production and efficiency.	Supporting Element: Quantity
No of Style Change	Frequency of style changes, impacting productivity, efficiency, and quality.	Supporting Element: Quantity
Basic KPI: Production		
Efficiency	Comparison of actual output to what could be achieved with the same resources.	Basic KPI: Production
Productivity	Achievement toward goals based on the relationship between inputs and outputs.	Basic KPI: Production
Availability	Percentage of scheduled time employees or machines are productive.	Basic KPI: Production

KPI/Metric	Description	Classification (Kang et al., 2016 / ISO 22400)
Performance	Amount of product delivered relative to available productive time.	Basic KPI: Production
Line Wise Sewing Efficiency	Efficiency of sewing lines, often linked to man-to-machine ratio.	Basic KPI: Production
Basic KPI: Quality		
Defect per Hundred Units (DHU)	Number of defects found per hundred units produced.	Basic KPI: Quality
Quality	Percentage of perfect or saleable products produced.	Basic KPI: Quality
Comprehensive KPI		
Overall Labor Effectiveness (OLE)	Measures workforce utilization, performance, and quality, reflecting labor's impact on productivity.	Comprehensive KPI
Overall Equipment Effectiveness (OEE)	Quantifies how well equipment performs relative to its designed capacity, considering availability, performance, and quality.	Comprehensive KPI

2.3.3. IoT-Plattforms

The IoT is known for producing large amounts of data and for the potentials to grow these amounts even more. Therefore a scalable software infrastructure that is needed. That is where IoT-Plattforms come into play [Turki et al., 2024]. The authors also mention that IoT-Platforms help accelerating the solution development "[...] by providing foundational capabilities, avoiding the need to implement low-level infrastructure."
[Asemani et al., 2019] further highlight the different capabilities that are typical for IoT-Platforms.

Connectivity and Device Management Through various communication protocols the platforms connect with the devices, enabling them to communicate with each other, manage device status and configurations, handle software updates and provide mechanisms for error reporting.

Data Storage, Management, Analysis, Visualization Through connections to databases they store large volumes of data often in the cloud or locally. Also further data processing and analytics through various methods as well as visualizations through dashboards are possible.

Development and Deployment Tools By providing APIs and SDKs the developers are enabled to further create custom applications.

Auditing and Payments The Platforms help to have an overview over the data or compute usage and the resulting costs.

Service Management By giving an oversight over parameters like resource consumption, data requirements and access, the user can monitor vertical as well as platform internal services. The platforms also enable the communication between services or combination of basic services to create new ones.

Integration Platforms can be integrated with each other, other data sources and the cloud.

Fog/Edge Computing IoT-Platforms often support distributed data processing and storage. This can lead to less traffic due to processing close to the data source. Faster transmission would be enabled therefore and reinforced due to shorter communication distances.

The researchers go on to reveal that while commercial platforms carry all of the mentioned capabilities, open source platforms are often focused on specific capabilities. Thus in implementation sometimes need to be combined to deliver a holistic IoT-Platform.

2.3.4. Differences between Relational and Timeseries Databases

In the paper written by [Türkoğlu et al., 2024] it is analyzed how relational databases and time series databases compare regarding speed and storage efficiency when used in Grafana. First they point out the use case for relational databases is for single time data points, which can be related to other data points over various tables. Therefore enabling complex queries involving joins, aggregations and multiple tables. They make sure the data is accurate and stays consistent. Time series data bases on the other hand are made for data points with a timestamp and large volumes of data. This makes them ideal for IoT applications and real time data analytics. They are optimized to enable high speed read and write operations as well as efficient data storage. The differences in query return time are already there with small amounts of data, but when scaling up the amount they become clearly visible as can be seen in the figure below.

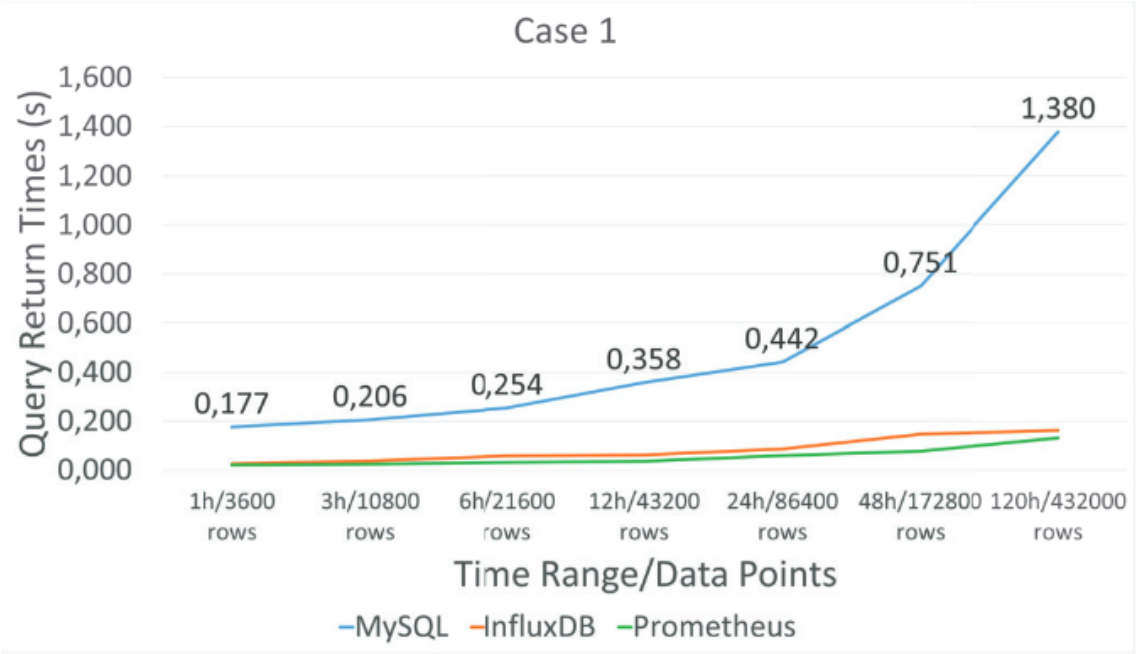

Figure 2.3.: Query Performance Relational vs. Time Series DB
(Turkoglu et. al, 2024, S. 2)

This is only one of three test cases but they all give a similar picture that shows the superiority of time series databases regarding query return times.

2.4. Legal Framework

Personal rights regarding personal data are highly important in Germany and the entire European Union. Therefore, it is important to provide an overview of the regulations that the proposed system must fulfill. According to the General Data Protection Regulation [REG,] of the European Union, it is illegal to automatically measure a worker's performance without human intervention that produces legal effects. Though this is not the intention of the system, if the produced data were traceable to a single worker, it could be abused to put pressure on the worker or find reasons to fire them if their performance is too low. The regulation also stipulates that workers must be informed if any personal data is being collected. In the case of the system in question, personal data could be produced in the form of amounts traceable to individual workers. Additionally, according to the German "Betriebsverfassungsgesetz § 87 Mitbestimmungsrechte," [87B,] the works council has co-determination rights regarding systems that measure worker performance.

3. Related Work

This section presents an overview of related solutions for sewing machines, with a focus on methods for data collection and processing in monitoring applications. It also examines general approaches to performance measurement in manufacturing. The analysis will likely address the types of metrics underlying these solutions, the sensor technologies used for data acquisition, and the strategies for data processing, storage, and analytics. This review aims to clarify the current state of the art, identify possible gaps in existing systems, and position this work within the field of automated performance measurement systems. Furthermore, the review seeks to highlight potential limitations and opportunities, including technical aspects such as hardware, software, and integration, as well as architectural considerations like standards, frameworks, and open-source approaches, which may influence the implementation of performance measurement systems.

In their seminal work, Jung et al. [Jung et al., 2020] propose a system for the analysis of sewing machine operator skill level and the complexity of the assigned sewing tasks. The present study bears a notable similarity to the aforementioned work, as both analyze the cycle time per unit. This objective is pursued by employing power consumption data of the sewing machines as the fundamental metric. To that end, a power monitoring system is employed, which is connected to the machines power plug without necessitating modifications to the sewing machine itself. Subsequently, the data is transmitted to a cloud server via wireless communication. The determination of quantity and time of work is achieved through the implementation of pattern analysis algorithms. The efficacy of the system is predicated on the non-intrusive nature of PMS. A notable disadvantage of this solution is the necessity of sufficient sample data to ensure the accuracy of the results.
Another system that uses power consumption measurement was proposed by Strazinskas [Stražinskas, 2025]. The utilization of current sensors enabled the discernment of fluctuations in motor current, which are inherently associated with the operational states of the sewing machine, including initiation, cessation, seam length, and stitch speed. The sensor is connected to an Arduino Uno microcontroller, which transmits the received data to a Raspberry Pi 3B+ microcomputer. The microcomputer is responsible for central data collection and storage. The data is stored in files, which is a disadvantage of the system because it is less efficient than storing it in an optimized time series database. Therefore, a greater quantity of space is required, and the rate of data processing is reduced. Additionally, the absence of any preprocessing mechanisms underscores the necessity for optimized data management, a crucial aspect that demands significant memory resources. Strazinskas's research indicates that the utilization of the present sensor engenders measurement errors and noise in the data.
Although adopting a divergent approach, the system proposed by Quoc et al. [Quoc et al., 2025] likewise relies on IoT devices (which were not further described) connected to the machine. The data of these devices is then transmitted to the cloud, where it is integrated with the enterprise's Management Information System (MIS). This integration facilitates the optimization of resource allocation. In light of the aforementioned data, a visualization is generated to illustrate the open positions of a contract and the extent to which it has been completed.
A divergent approach is posited by Wedanage et al. [Wedanage and Thiwanka, 2022] in their study "Fog Assisted Industrial Sewing," in which they utilize a mobile application to record the cycle

times of the sewing step.To initiate and conclude the sewing process on the workpiece, the operator is required to engage a designated button at the commencement and cessation of the operation. The data is structured in a JSON format and published via MQTT under a designated topic. A fog node, implemented using a Cisco IR829 industrial router, is subscribed to the specified topic and processes the data through a Java application. Additionally, the application utilizes a RethinkDB database, which is characterized as open-source and specifically engineered for real-time applications. The fog node is already capable of performing local data analytics. In the cloud, data from all fog nodes is combined to provide real-time alerts and analytics through dashboards. These dashboards visualize cumulative average cycle times per member and compare them to takt time (maximum time allowed to produce one product to meet customer demand). The employment of fog computing facilitates the system's notable scalability, a consequence of the distribution of computing tasks across multiple fog nodes. This configuration facilitates the system's capacity for real-time data analytics and reduced bandwidth utilization. A notable drawback of this system is its reliance on manual input from operators, which introduces a margin of error. Moreover, no historical analytics have been conducted, despite the existence of a foundation for such analyses. A notable drawback of the system in question, as well as those proposed by Jung et al. and Quoc et al., is the utilization of a cloud, which gives rise to several concerns, including but not limited to data security, latency, and ongoing costs. The merits of the aforementioned systems include their capacity for retrofitting older machines and their non-intrusive nature.

Beyond these closely related implementations, a broader perspective is provided by Tambare et al. [Tambare et al., 2022], who review various approaches to performance measurement systems. Initially, the discourse centers on two pivotal standards. The ISA-95 standard delineates entities at the shop floor level, where information technology systems such as ERP, CRM, cloud platforms, and SQL databases interact with operational technology components like sensors, actuators, microcontrollers, SCADA systems, and PLCs. Its primary function is to formalize production processes. Conversely, the ISO 22400 is employed for the formalization of performance metrics. The provided framework facilitates the definition, calculation, and dissemination of key performance indicators (KPIs), encompassing their contextual framework, formula, unit of measurement, range, and intended audience. The researchers then proceed to underscore the Scania case study, wherein these two standards are being utilized, as a paradigm of the practical implementation of international standards for performance measurement in a smart manufacturing context.
This case study by Samir et al. (samirKeyPerformanceIndicators2018) will now be examined in more detail. The system is composed of numerous autonomous, self-contained computational units that are each capable of performing independent computations and collaborating with each other. The software utilized for the processing and distribution of data has been developed using a Service Oriented Architecture (SOA) framework. This approach involves the implementation of multiple services that function independently of one another, with each service designated to specific responsibilities. The communication between the units is facilitated by an Enterprise Service Bus (ESB). The architecture further employs an Event-Driven Architecture (EDA), whereby events serve as triggers for services. This aims to address the issue of communication delays. The KPIs are designed in accordance with the ISO 22400 standard, and the IS-95 is employed to enable automated interfaces between enterprise and control systems.

Taken together, these studies illustrate a variety of approaches to IoT-based machine monitoring, each with specific strengths and limitations. In contrast to the systems examined in the aforementioned studies, the system presented in this thesis is designed to operate with a contemporary

sewing machine that facilitates data extraction directly from the machine. It is also intended to provide a more comprehensive overview of the production process by means of various Key Performance Indicators (KPIs), which will be addressed in subsequent sections. The system proposed in this thesis aims to incorporate the strengths of the discussed systems while mitigating their weaknesses. The system has been designed to align more closely with the proposed model in the case study by implementing standards and service-oriented software. Therefore, this facilitates enhanced scalability and maintainability.

4. Requirements Analysis

In order to select the most suitable technologies and design the system, it is first necessary to establish clear requirements. In this chapter, the aforementioned requirements will be enumerated and elucidated. In order to maintain consistency regarding the compliance level, the terms ”must,” ”should,” and ”will” were employed. The following words shall be explained in brief. The term ”must” is employed to signify an unconditional obligation, implying that the fulfillment of this requirement is not subject to discussion or negotiation. The term ”should” conveys a degree of desirability, indicating that fulfillment of the requirement would be advantageous. The term ”will” signifies that this particular requirement is currently under consideration for inclusion in the subsequent release. However, it is imperative to maintain awareness of this requirement so that the system can be designed in a manner that facilitates its seamless integration in the future.

4.1. Functional System Requirements

Requirement	**Explanation**
The system must show KPIs that are relevant for the sewing process	In the workshops there must be some KPIs that fit into the story of a textile production with a sewing process
The system should show aditional KPIs that are relevant for the manufacturing industry in general	Workshop participants are from all sorts of companies within the manufacturing industry
The system must present these KPIs in a visual manner that provides information about the classification of the current value e.g. with colors and thresholds	So that the management can act quickly upon the KPIs and does not need to lookup thresholds
The system must provide the user with the ability to change the timeframe on which the KPIs are calculated	Especially when looking at historic data it is usefull to be able to set the timeframe
The system should show graphs with historical data	Enables management to see trends and patterns

4.2. Non-Functional Requirements

Requirement	**Explanation**
The system must make use of open source software where possible	To be replecable by small and medium companies
The system must be capable to generate all of the KPIs from the machine-data without installing any additional sensors	
The system must be designed in a way that makes it easily scalable	Usually more than just one machine need to be considered for monitoring
The system must be deployable with minimal effort	To be able to deploy elsewhere with not much effort
The system must be capable to retrieve raw data via opc-ua	The PLC to which the sewing machine is connected publishes the data over opc-ua
The system should make use of existing patterns, frameworks and solutions were possible	The system shall serve as a reference for other systems that are more readily implementable
The system should update the KPIs in real-time (¡10s)	To support timely interventions
The system must be able to run on a local machine and therefore independent of any cloud service	
The system must provide the user with the ability to access the dashboard from within the shopfloor network	

4.3. IoT Platform Evaluation and Selection

The raw data from the PLC had to undergo processing, storage, and visualization. The utility of IoT platforms stems from their ability to execute these functions and frequently extend beyond them. These systems meet the criteria for scalability and leverage existing solutions. To ensure that the selected platform satisfies other requirements and does not conflict with any necessary ones, a decision support framework was developed. The present framework draws inspiration from the one proposed by Gustin and Jasperneite [Gustin and Jasperneite, 2022]. The researchers identified a set of criteria and classified them into categories. The categories encompassed ”Knock-Out,” ”Device Management,” ”Hosting,” ”General,” and ”Effort.” Each category was assigned a weight, with the exception of the ”Knock-Out” category. The ”Knock-Out” category indicates that a criterion tagged with it must be met; otherwise, the platform will not be considered further. The

weights are expressed as percentages, and when added together, they equal 100%. In addition, each criterion is assigned a distinct weight, thereby enabling the allocation of greater significance to specific criteria. The evaluation of each platform is subsequently determined by the following formula:

$$s_i = f_i \cdot 100P \cdot weight_{category} \cdot weight_{criterion}$$

In this context, f_i denotes the fulfillment factor. The newly developed framework is distinguished by the following characteristics: The criteria are flexible and should be altered according to the requirements. The aforementioned study concentrates on device management; however, this is not a salient issue in the context of the proposed system. The categories were modified to "Knock-Out," "High Importance," "Mid Importance," and "Low Importance." In this manner, the criteria are not constrained to a particular subject matter; rather, they can be assigned an extent of importance at will. The numerical values assigned to the categories are as follows: 1 for low importance, 2 for medium importance, and 3 for high importance. The new formula for calculating the platforms' score is more straightforward as well:

$$s_i = f_i \cdot weight_{category}$$

The fulfillment factor is typically binary in nature. The value of the variable can be either one or zero; however, in some cases, it can also be fractional. For instance, it would be advantageous for the platforms to support additional communication protocols beyond OPC UA. Specifically, the focus is on HTTP, MQTT, CoAP, and AMQP. In the event that a given platform offers support for only one of these, it is awarded a quarter point. In the event that this is the case, it will be elucidated in the subsequent explanation. The fulfillment of the criteria was determined in the same manner as that employed by the researchers. An investigation was conducted into the documentation, website, and GitHub repository of the platform. In the event that information regarding the availability of the criterion was ascertained, the point was granted. In the absence of pertinent information, the concept in question was deemed to be nonexistent. The subsequent paragraphs will provide detailed explanations for each of the aforementioned criteria. A selection of the materials was obtained from the publication by Gustin and Jasperneite if their utility was deemed to be beneficial. The remaining ones were developed in response to specific requirements.

4.3.1. Knock-Out Criteria

Availability of Documentation
The implementation, deployment, and utilization of the solution are contingent upon the availability of English-language documentation, which is imperative for effective operation. This criterion originates from the seminal contributions of Gustin and Jasperneite

Cloud independence
In order to guarantee that the platform can be hosted on a local server that is accessible via the shopfloor network and that no additional cloud services are required, this criterion was established. This criterion is derived directly from the stipulated requirements.

Ability to process raw data and derive KPI's from it
In order to visualize the KPIs, the data must first undergo processing, after which the KPIs are to be calculated based on the processed data. This criterion pertains to the requirement of calculating KPIs.

OPC-UA capability
The PLC that controls the sewing machine transmits the data via the OPC UA protocol. There-

fore, it is imperative that the platform demonstrate its capacity to effectively manage this particular protocol. This criterion is predicated on the requirements.

4.3.2. High Importance Criteria

Dashboarding Capabilities
In order to facilitate the visualization of KPIs, the platform must possess the capacity to construct dashboards. This criterion pertains to the necessity of possessing the capacity for visualizing KPIs.
Actively Maintained
This criterion is instrumental in ensuring the platform's security and compatibility with evolving technologies and standards.The necessity for this criterion arises from the imperative for open-source software.
Completion of Server SW
This criterion offers insight into the ease with which the software can be installed. A score of one is assigned if a Docker image is provided, and a score of zero is assigned if only source code is available. This criterion is derived from the work of Gustin and Jasperneite
Development of the server-side application
This criterion delineates the ease with which rules can be created and data processed within the application. In the event that a rule engine (low code) is provided, a point will be awarded. In the event that an SDK is provided, a total of 0.5 points will be allotted. This criterion originates from the work of Gustin and Jasperneite
Examples for Server-side implementation
These examples methodically illustrate the server-side implementation process and the anticipated outcomes.This approach has been shown to expedite the implementation process. This criterion originates from the work of Gustin and Jasperneite

4.3.3. Medium Importance Criteria

Supports MQTT, HTTP, CoAP, AMQP
The implementation of these protocols would significantly augment the system's scalability, as it would facilitate the connection of a greater number of machines, gateways, and microcontrollers. This criterion pertains to the necessity of having a scalable system with minimal effort.
Availability of Tutorials
Tutorials are instrumental in facilitating user comprehension of the platform and its applications, thereby accelerating the development process. This criterion originates from the work of Gustin and Jasperneite

4.3.4. Low Importance Criteria

Fault Detection
This feature facilitates the collection of data pertaining to anomalous behavior or fault conditions of the machine. Therefore, it facilitates the identification and resolution of potential issues. While it would certainly be a beneficial addition, this feature does not directly align with the primary objectives of the system. This criterion is derived from the work of Gustin and Jasperneite
Heartbeat Monitor
This feature periodically ascertains the online status of the connected machine. Consequently, this would facilitate the identification of device connectivity issues and ensure system reliability. This feature is commendable, yet it does not directly align with the primary function of the system.

This criterion is derived from the work of Gustin and Jasperneite

The platforms to be evaluated were derived from two papers that themselves evaluated IoT platforms. Initially, the open-source platforms from the aforementioned study were selected. These include openBalen, Thingsboard, Kapua, OpenRemote, Ubuntu Core, FIWARE, OpenMTC, Mainflux, and DeviceHive. In a separate study, Turki's 2024 investigation [Turki et al., 2024] focused exclusively on the evaluation of open-source Internet of Things (IoT) platforms, utilizing data derived from GitHub statistics. The following factors were taken into consideration: the stars that were given by users, health, contributors, open issues, closed issues, and releases. Due to the more objective nature of this evaluation, only the top five IoT platforms were selected. The restriction to a specific number was necessitated by the impracticality of evaluating an indefinite number of platforms, as this would have entailed an inordinate amount of time. The top five platforms identified in this study are Thingsboard, Digiot, Mainflux, OpenRemote, and IotSharp. However, it should be noted that only two of these findings were in disjunction with those selected from Gustin and Jasperneite's work. In the course of the research, the Node-RED platform emerged on a regular basis; however, it was not referenced in the evaluated documents. The platform was also mentioned by a colleague, and it was therefore given due consideration. Additionally, UMH was selected for evaluation because it builds upon robust open-source tools, such as Grafana and Apache Kafka, and supports a wide array of protocols.
The ensuing table 4.1 presents the results of the aforementioned calculations. It is imperative to note that only the platforms that met all of the Knock-Out criteria are displayed therein.

Table 4.1.: IoT Platform Evaluation

IoT Platform	**Score**
Thingsboard	19.5
Node-Red	21.5
FIWARE	19.0
OpenRemote	12.0
UMH	20.5

The evaluation metrics for Node-RED, ThingsBoard, and UMH exhibited a high degree of similarity, suggesting comparable levels of suitability among these options. Consequently, an in-depth investigation was conducted to ascertain their system requirements. Of the three options, the UMH configuration exhibits the most demanding specifications. It requires 200 GB of persistent memory, 16 GB of RAM, and an 8-core CPU [UMH,], all of which the server configuration is incapable of meeting. The server configuration, on the other hand, has a specification of 100 GB of persistent memory, 8 GB of RAM, and an 8-core CPU. In the context of Thingsboard, the minimum amount of RAM required is specified as 4 GB [thingsboard,]. According to the official

Node-RED website, there are no listed system requirements. However, it has been documented that it is capable of operating on a Raspberry Pi [Run,]. Additionally, certain users have attested to the successful execution of the program on a Raspberry Pi 2 model [Nod, 2020], which possesses a mere 1 GB of RAM and a 900 MHz quad-core central processing unit [Ltd,]. Given its ostensibly minimal complexity and superior performance metrics in the preliminary evaluation, Node-RED was designated as the IoT platform for the system under development.

4.4. Database Selection

The selection of a suitable DBMS was imperative for the storage of the data. In the theoretical foundations, the rationale for the superiority of a time series database in the context of IoT streaming data over a relational database was previously delineated. The selection of the Timeseries Database Management System (TS-DBMS) was executed with a lower degree of sophistication compared to that of the IoT Platform. The rationale underlying this matter is predicated upon the temporal limitations imposed on the undertaking. A comprehensive investigation was conducted to ascertain the most prominent TS-DBMSs on db-engines.com, a website dedicated to determining the popularity of DBMSs. The following metrics have been identified as contributing factors to the popularity ranking:

- The number of mentions on websites
- General intrest in the system using Google Trends
- The number of related questions and posts on well-known IT Q&A sites
- The number of job offers in which the system was mentioned
- The number of mentions on professional networks
- The number of mentions on social networks

Subsequently, the capacity of InfluxDB to process boolean values was ascertained. This assertion was corroborated in the documentation [Wri,]. In addition, an assessment was conducted to ascertain whether InfluxDB is compatible with Node-RED. The package that was found (node-red-contrib-influxdb 0.7.0) has been determined to only support InfluxDB 1.x and 2.0 [Nod,]. Prior to this, it was determined that InfluxDB 3 Core's capacity for querying data is confined to a span of merely 72 hours [Que,], a limitation that was deemed inadequate in view of the necessity to compute historical KPIs. Consequently, InfluxdDB v2 was selected due to its compatibility with Node-RED and its capacity to support unlimited querying windows.

4.5. Dashboarding Tool Selection

Initially, the objective was to leverage the dashboarding capabilities of the selected IoT Platform Node-RED. However, it appears that the utilization of this particular node-RED feature for visualization purposes is not a prevalent practice. In contrast, another software program, Grafana, was utilized in a significant number of cases similar to the one under consideration in this study. Consequently, the decision was made to utilize Grafana for the development of dashboards, with the objective of facilitating the visualization of the KPIs. This approach was adopted to enhance the probability that other colleagues would be acquainted with the solution, thereby facilitating a

more streamlined handover process. The decision was further corroborated by Rani in his work [Rani, 2025], who states that it is highly appropriate for time series visualization and monitoring. Additionally, the integration of the system with various databases, including InfluxDB, is emphasized.

4.6. System Architecture

The system architecture was designed in accordance with the three-tier architecture pattern from the IIRA mentioned in the foundation chapter. The Three Tier Architecture Pattern was selected because it exhibits a clear separation of concerns between the edge, platform, and enterprise tiers, thereby facilitating enhanced maintainability. Additionally, the progression of the distinct components is rendered more straightforward and less susceptible to errors in comparison to a monolithic application. The selected architecture pattern offers a distinct advantage, namely the seamless integration of supplementary devices and services. The incorporation of devices into the Node-RED system is contingent upon their connectivity to the shopfloor network. The integration of services at the platform or enterprise tier can be accomplished without the need for significant alterations to the existing system, thereby facilitating flexible expansion and adaptation to novel requirements.

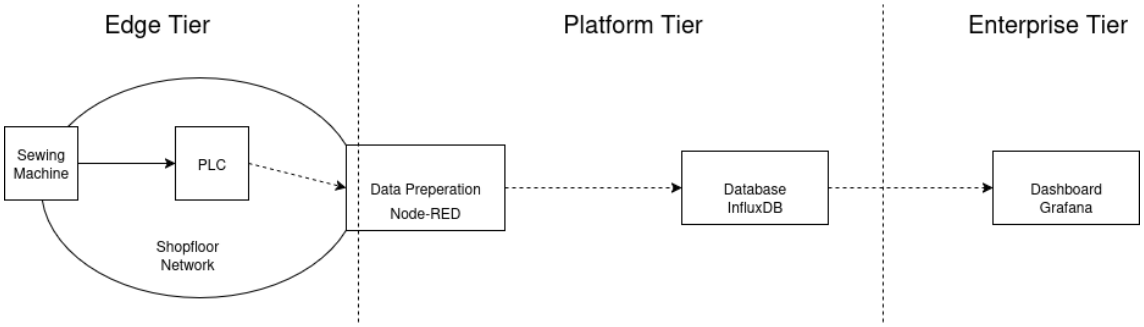

Figure 4.1.: System Architecture

The signal outputs from the sewing machines are wired to the PLC, as illustrated by the continuous line. Conversely, the dashed lines signify the transmission of data either over a network or within a server. The PLC provides these signals via OPC UA, and it is accessible via the shopfloor network. In the context of this architecture, Node-RED performs a variety of functions. Initially, it functions as an IoT gateway, thereby aggregating data from disparate sources. In this particular instance, the sole source of data is from the PLC. However, the component is also capable of performing data processing operations. Subsequently, the data is transmitted to the database. From there, it can be queried with the dashboarding solution.

4.7. KPI Selection and Justification

The selection of KPIs is contingent upon the fulfillment of two requirements. First, it is necessary to establish a set of KPIs that are pertinent to the sewing process. Secondly, it is imperative to establish KPIs that are pertinent to manufacturing companies in general. Additionally, it is imperative that these KPIs be computable based on the available signals. Therefore, the initial step was to ascertain which elements that provide support can be calculated based on the signals. In addition, a determination was made regarding the observations that can be derived from the

available signals. It is imperative to acknowledge that merely four of the machine's signals are accessible.
Based on the analysis of the available machine signals—namely, thread trimming, pressure foot position, and main shaft rotation—a distinct production pattern can be identified and quantified. The observed sequence is as follows:
Initially, the pressure foot is lifted to allow the operator to insert the material. Subsequently, the pressure foot is lowered, securing the material in place. The sewing process then Beginnt, as indicated by the activation of the main shaft rotation signal. This signal remains true for the duration of the seam and returns to false upon completion. Following this, the pressure foot is raised again to facilitate removal of the finished workpiece. During this interval, the thread trimming signal is activated, indicating that the thread is being cut. Finally, the pressure foot is lowered once more, marking the conclusion of the production cycle.
According to the established sequence of events, the duration of the machine's operational time can be determined. The duration of the main shaft rotation signal being true corresponds to the actual sewing time. The number of pieces produced can be determined by counting the instances in which the thread trimming signal transitions to true, as each event signifies the completion of a workpiece. Furthermore, the cycle time can be calculated by measuring the interval between consecutive activations of the thread trimming signal. The signal "other than homescreen" could be indicative of a modification in the machine's configuration. The signal "homescreen and not sewing" can be used to observe the machine's activation. One might hypothesize that this phenomenon is merely an inverted state of the "upper shaft rotating" signal. This phenomenon is observed when the machine is activated. However, when the machine is deactivated, all signals are false. Therefore, the transition from all signals being false to "homescreen and not sewing" being true, while the other signals remain false, would indicate the activation of the machine. This could be used to measure setup time until the first pattern arises. When only the "homescreen and not sewing" signal is active, it means the machine is idle. Therefore, the length of time that the machine is in the idle state can be measured.

4.7.1. Supporting Elements

To facilitate a comprehensive understanding of the supporting elements and the KPIs, the following section provides detailed explanations of these elements, along with their respective formulae if the calculation involves more the simply counting or measuring. Most of the definitions stem from Kang et al. [Kang et al., 2016] Hierachical Structure of Key Performance Indicators [...].

Processed Quantity (PQ) The production of a piece is initiated with the trimming of the thread. The calculation of the pieces is achieved through the enumeration of these events.
Planned Downtime (PDT) In accordance with standard practice, the planned downtime is intended to encompass deliberate maintenance procedures and related activities. However, in the absence of such scheduled maintenance, the only components that are considered to be within the scope of planned downtime are the designated breaks. While it does not technically constitute a form of downtime, for the sake of illustration, the aforementioned period will be designated as "downtime."
Planned Operation Time (POT) The designated period when a machine is available for use.
Actual Down Time (ADT) The actual duration during which production is suspended due to breakdowns, brief stoppages, and other unanticipated disruptions.
Actual Production Time (APT) The period during which the machine is actively producing for

an order, encompassing only value-adding activities. In the instance of the production pattern, the worker is engaged in the active utilization of the machine. The calculation of production time is determined by the summation of the duration of these patterns.
Planned Cycle Time (PCT) The estimated ideal time required to manufacture a single unit. The cycle time is defined as the time interval between the completion of one task and the commencement of the next. In this work the cycle time is only measuring the single step undertaken on the sewing machine under consideration. Kang et al. designate it ”planned run time per item.” However, given the significance of this metric for sewing-specific processes, it is referred to as ”cycle time” in accordance with sewing-related publications.
Actual Cycle Time (ACT) The actual time required to manufacture a single unit.
Setup Time (ST) This period is typically characterized by the machine's preparation for the subsequent order. In this study, the setup time is defined as the interval between the initiation of the machine at the commencement of a shift and the first production of a piece. This specific definition is employed due to the inherent limitations of the measurement capability, which restricts the evaluation to this particular temporal interval.
Idle Time (IT) The time during which the machine could potentially be utilized but is not being actively employed.
Planned Busy Time (PBT) The scheduled period during which a machine is actively engaged in operation.

4.7.2. Maintenance Elements

Time to Repair (TTR) The period during which a machine is rendered inoperable due to a malfunction, i.e., undergoing maintenance or repair.
Time to Failure (TTF) The period during which a machine is operational, commencing at the conclusion of the repair process and concluding when the machine experiences a subsequent failure.
Operating Time between Failures (OTBF) The effective production time of a unit measured between two successive instances of machine failure.
Failure Events (FE) The total count of occurrences during which the machine was non-operational.

4.7.3. Basic KPIs

The supporting elements that have been explained are used to calculate the basic KPIs. Some of these KPIs were mentioned in studies on sewing performance measurement. These are performance, availability, OEE and average cycle time. However, these KPIs are not only interesting for the textile industry because they are also generally relevant to production. Some other KPIs are simply averages of supporting elements. They help identify the reasons behind the values of higher-level KPIs. Each of the selected KPIs will now be explained and shown with its respective formula.

Performance
The performance rate is a KPI which is necessary to calculate overall equipment effectiveness (OEE). It measures the speed at which equipment operates. It is compared to the equipment's maximum operating speed because this comparison is necessary to determine the equipment's

effectiveness. [Muchiri and Pintelon, 2008]

$$\text{Performance Efficiency (P)} = \frac{\text{Theoretical Cycle time (hrs)} \times \text{Produced Pieces}}{\text{Operating Time (hrs)}}$$

The term ”actual output” refers to the number of units that are produced during the operation, whereas ”theoretical cycle time” signifies the minimum potential cycle time that could have been achieved at the fastest practical speed. Operating time is defined as the duration during which the machine is in active operation.

Availability is defined as the ratio between the actual production time and the planned busy (production) time.

$$\text{Availability} = \frac{\text{Actual Production Time}}{\text{Planned Busy Time}}$$

Quality This KPI is defined as the proportion of high-quality products in relation to the total number of units produced.

$$\text{Quality Rate (Q)} = \frac{\text{(Total Production - Defect Amount)}}{\text{Total Production (Units)}}$$

Average Cycle Time The cycle time for each produced unit is first aggregated to create a sum. This sum is then divided by the produced quantity to calculate the mean cycle time.

$$\text{Average Cycle Time} = \frac{\text{Total Cycle Time}}{\text{Produced Quantity}}$$

Average Setup Time The calculation of the setup time is performed by aggregating the values across all shifts and subsequently dividing this sum by the total number of shifts.

$$\text{Average Setup Time} = \frac{\text{Total Setup Time}}{\text{Nr. of Shifts}}$$

Average Idle Time The aggregate idle time is subsequently divided by the total number of shifts within the specified timeframe, thereby yielding the average idle time per shift.

$$\text{Average Idle Time} = \frac{\text{Total Idle Time}}{\text{Nr. of Shifts}}$$

Mean Time to Repair The cumulative repair time is subsequently divided by the total number of failure events, thereby yielding the average repair time per event.

$$\text{Mean Time to Repair} = \frac{\text{Total Time to Repair}}{\text{Failure Events}}$$

Mean Time to Failure The cumulative time to failure is subsequently divided by the total number of failure events, thereby yielding the average time to failure per event.

$$\text{Mean Time to Failure} = \frac{\text{Total Time to Failure}}{\text{Failure Events}}$$

Mean Operating Time between Failures The cumulative operating time between failures is subsequently divided by the total number of failure events, thereby yielding the average operating time between failures per event.

$$\text{Mean Operating Time between Failures} = \frac{\text{Total Operating Time between Failures}}{\text{Failure Events}}$$

Average Actual Operating Time The aggregate actual operating time is subsequently divided by the total number of shifts within the specified timeframe, thereby yielding the average actual operating time per shift.

$$\text{Average Actual Operating Time} = \frac{\text{Total Actual Operating Time}}{\text{Nr. of Shifts}}$$

4.7.4. Comprehensive KPIs

According to Kang et al., based on the Basic KPIs, comprehenisve KPIs can be calculated.

Overall Equipment Effectiveness (OEE) Overall Equipment Effectiveness (OEE) is formally defined as a comprehensive metric quantifying the total performance of a given piece of equipment. It integrates factors of availability, performance, and quality to provide a holistic assessment of operational efficiency. Bluntly said "[...]the degree to which the equipment is doing what it is supposed to do ." [Muchiri and Pintelon, 2008]

$$\text{Overall Equipment Effectiveness} = \text{Availability} \times \text{Performance Efficiency} \times \text{Quality Rate}$$

Net Equiptment Effectiveness (NEE) Analogous to OEE, this methodology incorporates setup time through the modification of the availability KPI, whereby it is redefined as the ratio between actual unit processing time (AUPT) and planned busy time (PBT). The AUPT encompasses the temporal duration during which the machine is actively utilized for order execution, inclusive of the requisite setup time for the respective order. Because in the setting of this work there are no orders, it is not possible to calculate the AUPT. But it is approximated by taking into account the setup time for a shift. The original formula for the NEE is as follows.

$$\text{NEE} = \frac{\text{AUPT}}{\text{PBT}} \cdot \text{PE} \cdot \text{QR}.$$

By changing the first factor into simply adding the actual production time and the setup time, the approximating formula results into:

$$\text{NEE} = \frac{\text{APT} + \text{ST}}{\text{PBT}} \cdot \text{PE} \cdot \text{QR}.$$

5. Implementation

5.1. Sewing Machine Signal Connection and Data Acquisition

The final signals necessary to calculate all selected KPIs are ”Thread Trimming,” ”Pressure foot,” ”Upper shaft rotating,” and ”Main menu and not sewing.” The retrieval of these signals was contingent upon their initial assignment to the correct output pins in the machine’s menu configuration. The subsequent challenge was to establish a connection between the output pins and the input pins of the WAGO PLC.
A connection for one signal from the sewing machine had already been established during an earlier project, but unfortunately, the connection was inadequately documented. This proved to be more confusing than helpful. The connection was implemented as follows: The signal pin of the sewing machine was connected to the 0V reference of the PLC. The 24V pin of the sewing machine was connected to the signal input connector of the PLC. Figure 5.1 provides a visual aid to facilitate comprehension of this configuration. This arrangement made sense once it was discovered that the sewing machine signals are NPN type, while the PLC exclusively accepts PNP signals as input. Typically, when the sewing machine would also have signals of type PNP, the signal pins would simply be connected to the signal connectors of the PLC. Concurrently, the 0V connector of the PLC would be connected to the GND pin of the sewing machine.

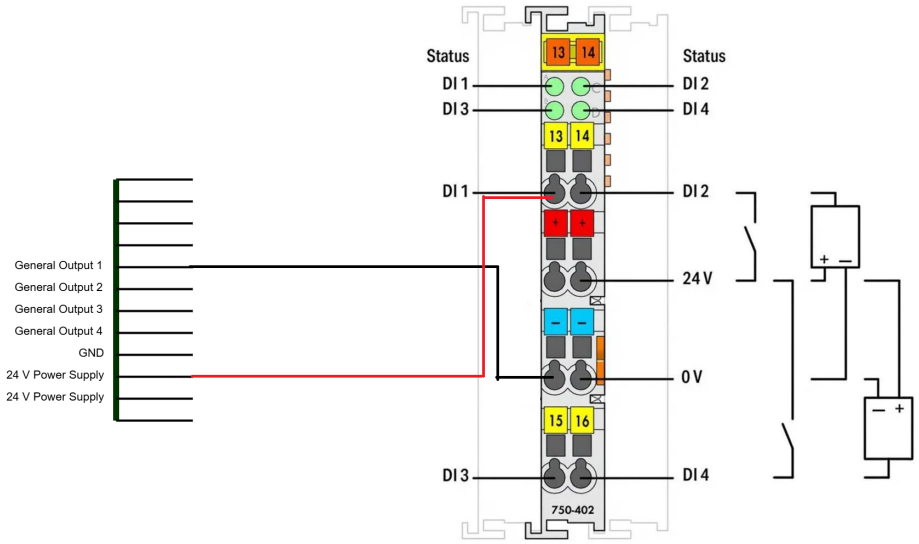

Figure 5.1.: Initial Connection of Sewing Machine to PLC

Note: The 12-pin connector of the sewing machine is located on the left side. This connector contains various pins, including those that function as signal pins. The left side of the module contains the PLC input module. The DI1-4 marked connectors of the PLC serve the function of signal input connectors.

The system was configured so that when a signal becomes active, it draws current through the signal input connector of the PLC, resulting in a high signal. However, this implementation was limited to only two signals, because there is only one 0V connector available for two signal input connectors. In total, there are two 0V connectors and four signal input connectors. When two 24V

pins are connected to one 0V connector via the signal input pins, an active signal draws current through the 0V connection and simultaneously pulls current from both input signal connectors. This results in invalid signals.
The resolution of the aforementioned issue necessitated the conversion of the NPN signals of the sewing machine into PNP signals, which are compatible with the PLC. For the execution of this task, an optocoupler was utilized. The implementation was executed in accordance with the subsequent description. Subsequently, the signal output pins of the sewing machine were connected to the signal input connectors of the optocoupler. Furthermore, a connection to the 24-V power supply of the sewing machine was established for each signal input. The signal connectors on the output side of the optocoupler are connected to the signal input connectors of the PLC. Concurrently, the 24-V power supply of the PLC is connected to the VCC input connector of the optocoupler. Additionally, the 0V connector of the programmable logic controller PLC is linked to the GND connector of the optocoupler. The wiring of the optocoupler can be observed in the following Figure.

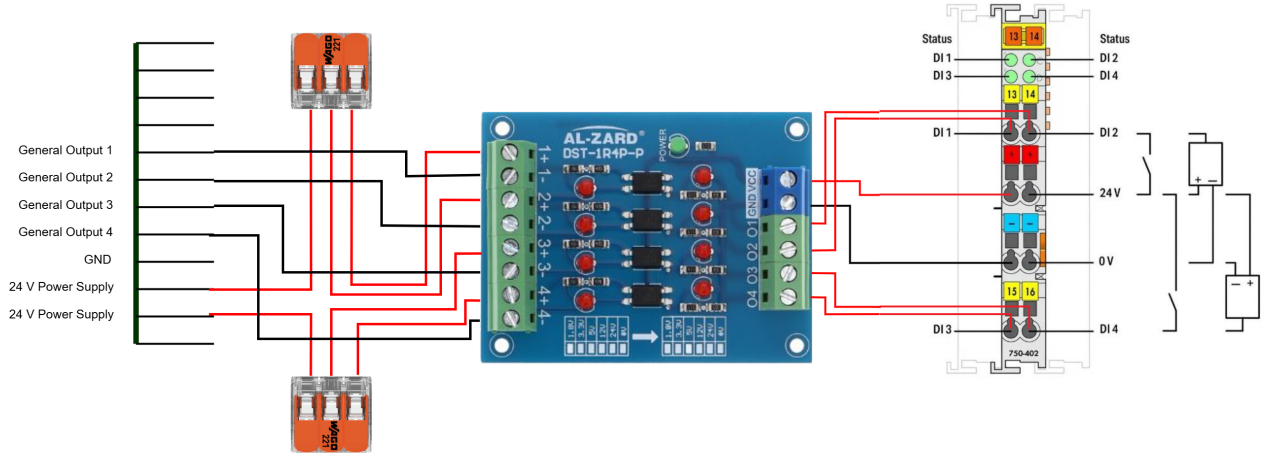

Figure 5.2.: Initial Connection of Sewing Machine to PLC

As illustrated, the red cables represent electrical connections originating from or terminating at a 24V power source. Additionally, the white cable located on the left side is linked to the 24V source of the PLC. The black cables serve as connections to ground or to the signal pins on the right side. These connections function in a manner that pulls down the current when signals are in a state of activity. In order to ensure the signal's availability across the shopfloor network, it was imperative to program the PLC in a manner that facilitated this objective. The input signals simply needed to be assigned to a value and then published over OPC UA.
Initially, only the IP address and the OPC UA port of the PLC were known. In order to enhance comprehension regarding the retrieval of data from the aforementioned system through the utilization of an OPC UA client, the development of a Python script was undertaken. The script under consideration took the two givens, established a connection, and navigated through the OPC UA server's node structure. The search is conducted for a "DeviceSet" node, which is understood to generally contain industrial devices, such as sewing machines. For each device identified, an exploration of its variables and child objects is initiated. It is important to acknowledge that, at this juncture, the connection from the preceding project was still in place. Further exploration was necessary to ascertain the nature of the connection and to identify any additional machines that were connected to the PLC. At this time there was a delay in communication with "Brother Internationale Industriemaschinen GmbH." Therefore, the necessary information regarding the

configuration of the sewing machine signals was not available. A decision was reached to initiate an exploration of effective communication methods with the OPC UA server.
Subsequent to the establishment of a connection to the PLC for all signals, a diagnostic procedure was conducted to ascertain the availability of all signals. For this purpose, a tool known as UAExpert was utilized. The software under discussion is an OPC UA client that provides a user interface for development, testing, and monitoring. The device was utilized for the purpose of monitoring the values of the signals. Consequently, the actions that were expected to elicit the signals were executed on the sewing machine. Initially, the process was proceeding according to plan.

5.2. Data Preprocessing

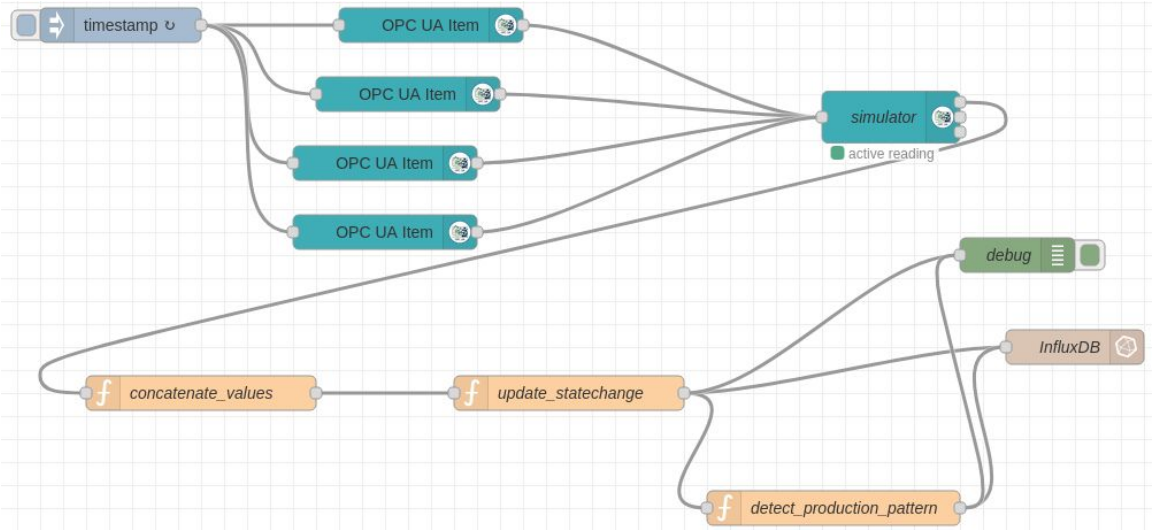

Figure 5.3.: Data retrieval and preprocessing

In the context of this study, Node-RED was utilized as a service to retrieve data that had been published by the PLC's OPC UA server. This feature facilitates the preprocessing and subsequent injection of data into the database. The retrieval of data from the OPC UA server is facilitated by an OPC UA package known as "node-red-contrib-opcua." The node designated as "timestamp" at the inception of the flow serves merely to initiate the flow. The temporal configuration of this feature may be set to various intervals or fixed times. The decision was made to establish an interval of 0.2 seconds. This interval was selected due to the fact that some operations on the sewing machine have an execution time of approximately one second when operated by inexperienced workers. In light of the dearth of seasoned professionals, it was deduced that an experienced worker would exhibit a fourfold increase in efficiency. To ensure the capture of all events, the interval was set to 0.2 seconds, equivalent to the execution time of a worker operating at a rate five times faster. Subsequent to the completion of the trial, the interval may be recalibrated in accordance with the findings of the evaluation. The four subsequent nodes, designated as OPC UA Items, each contain the namespace and ID of the various signals. These are subsequently fed into the simulator node. The node in question has been configured with the endpoint, which consists of the IP address of the PLC, the port, and the name of the OPC UA server, if such a server exists. This node is responsible for generating the values for each of the OPC UA items in a serial manner. The succeeding node is one of three function nodes. These function nodes contain JavaScript code that executes three distinct preprocessing steps. The initial step involves the concatenation of the four

values into a single string. The primary rationale for concatenating values is that, in InfluxDB, each value is assigned its own table. In the event that a query is executed over a set of values, the tables must be joined, a process that is computationally intensive. This phenomenon can result in substantial delays in the final dashboard. The concatenation of values eliminates the necessity for joining tables, as all values are subsequently stored within a single table. This preprocessing step confers an additional advantage, namely that it facilitates the subsequent step in the procedure. The subsequent preprocessing step in the function node designated "update_statechange" involves the verification of whether the concatenated string undergoes alterations. A modification in the configuration of the string is indicative of a state alteration in the sewing machine. Consequently, the string undergoes an output transformation. The initial concatenation of the values reduces the number of comparisons from four to one string. This preprocessing step was implemented to reduce the amount of data stored in the database. In previous iterations, this step was omitted, resulting in a substantial reduction in query processing speed relative to subsequent iterations. The output of the second preprocessing node is comprised of four separate key-value pairs of the signals, as well as these in a concatenated string format. It is important to note that these values are only provided as output when there is a change in one of the values compared to the previous state. The output of this node is utilized as an input for the third preprocessing node. In the preceding section (4.0.7), the production pattern was delineated as a metric for quantifying the duration of machine utilization for value-adding operations. The final preprocessing node, designated "detect_production_pattern", serves to ascertain the presence of this pattern. The output of this function is the key "pattern" with a value of true, indicating the beginning of the pattern, or false, indicating the end of the pattern. Additionally, the timestamp indicating the end or beginning of the pattern is included in the output. Upon completion of the data flow, both outputs from the "updateStateChange" function and the "detect_production_pattern" process are transferred to the designated InfluxDB node as input. This node has been configured with the address and the API key of the Influx database. The component in question is an InfluxDB out node, which signifies that it is capable of writing data exclusively to InfluxDB. This node stems from the node-red-contrib-influxdb package. In the course of each write operation, the system automatically incorporates a timestamp into the input. However, it is imperative to incorporate a timestamp at the "detect_production_pattern" node, as the pattern's detection is only possible in a retrospective manner. Consequently, the timestamp of the InfluxDB node would be inadequate.

5.3. Data Storage and Post-Processing

In the context of InfluxDB, it is necessary to establish a designated bucket for the designated project. In the context of a relational database management system (RDBMS), a bucket can be regarded as analogous to a schema. The bucket is configured with a retention policy and access rights. The retention policy is a mechanism that governs the duration for which data is retained in the bucket. The preliminary step in the process was the creation of an initial bucket, into which data was subsequently streamed from Node-RED. Initially, an unlimited retention policy was configured for the specified bucket. Subsequently, in light of apprehensions pertaining to an excess of data, the determination was made to curtail the retention policy to a span of one day. Following the initialization of the process, the data of the first bucket is aggregated and stored in a separate bucket that employs an unlimited retention policy. The aggregation process leads to a substantial reduction in the amount of data that must be stored. Aggregation is performed within designated tasks. These tasks can be configured to execute either upon the completion of a specified time interval or through the utilization of cron syntax, which facilitates the establishment of

a predetermined time for regular aggregation. The latter is employed in this implementation. The rationale underlying this necessity is that it is imperative for the aggregation to occur during the period between the conclusion of one shift and the commencement of the subsequent one. This phenomenon can be attributed to the prevalence of data aggregation processes that are contingent upon the identification of the initial and concluding points of an event. In the case that the aggregation occurred between these events, the result would be a falsification. The aggregation could also be executed at eight-hour intervals, thereby aligning with the duration of a typical shift. However, given the legal framework delineated in the foundational chapter, the daily aggregation is applied. The implementation of an eight-hour aggregation period would result in the accumulation of data that could be traced back to a specific worker. The aggregation of data over the course of an entire day encompasses three shifts. Consequently, it is not possible to draw any conclusions from the data regarding the individual performance of a worker. A notable benefit of this approach is a significant reduction in the necessary storage capacity, amounting to approximately two-thirds of the original requirement. The aggregation of these data is facilitated by the utilization of the flux query language. Throughout the developmental process, the aggregations were meticulously formulated in flux notebooks. These notebooks bear a resemblance to Jupyter notebooks and can be utilized within the web user interface of the InfluxDB DBMS. Cells for flux queries and additional cells for visualizing the result or displaying the resulting table(s) can be added to the notebook. In this manner, the proper functioning of the query can be readily corroborated. A thorough exposition of the methodology employed for the verification of results can be found in the Evaluation chapter. The results of the aggregations encompass all of the supporting elements.

5.3.1. Flux Query Language

Listing 5.1: Flux aggregation query for produced quantity

```
from(bucket: "sewing-machine-1-simulated")
|> range(start: -24h)
|> filter(fn: (r) => r["_measurement"] == "sewingMachine1")
|> filter(fn: (r) => r["_field"] == "thread_trimming" and r["_value"]
    == true)
|> count()
|> map(fn: (r) => ({r with _time: now()}))
|> map(fn: (r) => ({r with _field: "pieces"}))
|> to(bucket: "simulated-tasks")
```

A comparison of the flux query language with SQL-like query languages reveals notable distinctions in their respective syntax and semantics. Accordingly, the fundamental principles of flux querying are elucidated herein. The initial step in the process is the selection of the bucket from which the data is to be utilized for processing, as illustrated in the initial row. The ”—¿” (pipe forward) symbol is employed in each subsequent row. This operator indicates that the result of the preceding operation is to be utilized in the subsequent operation. The subsequent row is designated for the specification of the timeframe for the desired data. This is a common practice at the outset to minimize data and, consequently, processing time. Furthermore, the subsequent filtration operations function in a similar manner, thereby allowing for the selection of only the pertinent data. The ”_measurement” constitutes a logical grouping of the data, encompassing numerous tags and fields. The utilization of tags facilitates the creation of additional data groups. For instance, in the event that a project encompasses multiple sewing machines, the measurement name could be altered to ”State,” and the tag could contain the names of the identifiers for each sewing ma-

chine. Fields are defined by a name and a value, and their distinguishing characteristic is that they undergo changes over time. Subsequent processing of the data typically occurs subsequent to the filtration stage. The number of operations that can be chained together is unlimited. The count operation in this particular instance is one of numerous aggregate functions that reduce the quantity of values to a single value. The map function can be used to apply a function to every row in a data set. Upon completion of the query, the final result is returned. In this particular instance, the result is written to an alternate bucket.

5.3.2. Postprocessing Task Queries

This subsection briefly explains how aggregations of all supporting elements are queried.

Processed Quantity

The sum of all thread trimming events where the value is true is being calculated.

Actual Downtime

Downtime events are determined by summing idle or machine-off times that exceed a set threshold. Keep in mind that sometimes a worker goes to the bathroom or takes a short break. Initially, the threshold was set to ten minutes. Then, it is filtered for all events where all signals except "main_menu_not_sewing" are false. The latter can be true (idle) or false (machine off). All remaining entries above the threshold are summed up.

Actual Production Time

The sum of all event durations where the pattern tag is true is being calculated.

Actual Cycle Time

The duration between two cycle times is measured and summed up. To ensure that no breaks or downtime events are included, the same threshold used for the downtime measurement is applied to filter the values.

Setup Time

The time from when the machine is turned off until the first sewing event is measured and summed up for each shift of the day. A machine is considered off when all signals have the value false. A sewing event is characterized by only the "sewing_active" signal being true.

Idle Time

The duration of all events in which all signals except "main_menu_not_sewing" are false is measured and summed up. Then, the duration of the break is subtracted. Of course, these events could still contain downtime events. However, downtime is simply subtracted in later queries.

Failure Events

All events in which all signals are false, except for "main_menu_not_sewing," which can be either true or false, are filtered out. Then, the threshold filter is applied again. Finally, the number of events is counted.

The remaining maintenance elements were omitted because they depend on some of the supporting elements, such as actual downtime and production time. None of the supporting elements that are "planned" were mentioned because they cannot be queried, only defined.

5.4. Synthetic Data Generation

The implementation follows a staged approach to data generation and validation. Phase 1 employs offline synthetic datasets to design KPIs and to verify computations against ground truth. Phase 2 uses an online simulator via an OPC UA mock to evaluate streaming behavior, end-to-end

latency, and storage characteristics. Phase 3 targets on-machine integration. This approach decouples progress from hardware availability, enables controlled experiments with ground truth, and preserves reproducibility through fixed seeds and configurable scenarios. For phase 1, a Python script was developed. At the inception of the script, a series of parameters were established. The parameters in question encompass the duration of the shift, the number of hours that elapse before the commencement of the break, the duration of the break itself, the timeframe of the simulated data, and the step width with which the data is generated. In an effort to emulate the actual production process, it was determined that random values would be employed for various parameters, including setup times, idle times, sewing times, breakdown durations, and the number of breakdowns. For these random values, upper and lower limits were established at the outset. The data that was generated has the following structure. A period of inactivity occurs between each shift, during which the worker is responsible for setting up the machine and readying the workplace. This segment of time is considered to be ”idle.” Subsequently, the pattern delineated in Chapter 4.7 is reiterated in sporadic intervals. An idle event occurs between each step of the pattern because, under normal circumstances, the time frame for observing the pattern on a millisecond time scale invariably includes a period of inactivity between each action. Occasionally, intervals of downtime coincide with the patterns. Also one lunch break is included in each shift. In the script, the initial step entails the creation and subsequent storage of the data as a CSV file on a daily basis. Subsequently, it undergoes a transformation into the InfluxDB line protocol, and is then written in batches to the database. In the second phase, two Python scripts were implemented. The first is nearly indistinguishable from the script from phase 1. The generated data is almost identical to that of the original, with only minor variations, and is subsequently written to a CSV script. A salient distinction is that the data is populated to emulate a continuous stream of data analogous to that of the sewing machine. The filling up is realized through utilization of the forward fill method. This method utilizes a two-step process to address the aforementioned issues. First, it employs a technique that utilizes the previous state of the system to fill in the missing data. Second, it generates values based on the sampling rate. The generation of a new CSV file occurs at the 24-hour interval, thereby encompassing the subsequent 24-hour period. To ensure a sufficient supply of data, the process of data generation is initiated several hours prior to the timestamp of the most recent data entry within the previous 24-hour period. The second script utilizes the CSV file, initiates an OPC UA server, and streams the data at the designated sampling rate. The construction of the system is such that initiation is permitted at any point in time. The efficacy of this method is predicated on its ability to identify the entry that is temporally closest to the current time.

5.5. Infrastructure Automation and Service Health

The final system was designed to operate on a server equipped with the Fedora Linux operating system. In order to ensure a seamless deployment experience, particularly for colleagues who may repeatedly utilize the solution, it was determined that Docker should be employed as the orchestration tool of choice. A favorable consequence of this was that the development of the solution could be readily executed on various machines. The orchestration process was facilitated by the implementation of a Docker Compose YAML file. With the exception of the ”simulator,” for each service, the data and config folders were mounted to the data and config folders, respectively, within the project folder. This modification enabled the versioning of contents within folders. In the context of Node-RED, the InfluxDB API token was configured due to its transient nature when configured within the web user interface. A health check was configured for InfluxDB due to its propensity for crashing. Upon exiting, the container is restarted. A watchdog script has been

implemented to facilitate the exit of the container when it becomes unhealthy. It has been established that Grafana and Node-RED are contingent upon InfluxDB, necessitating a delay in their initiation until InfluxDB is operational. The fourth container is the simulator. The system under consideration simulates real-time data flow from the sewing machine. Due to the composition of two scripts, an additional supervisor configuration was implemented to orchestrate the order of execution. This necessity arises from the interdependence between the two scripts. Given the necessity of additional packages in Node-RED, it is imperative to execute a command during the deployment process to facilitate the installation of these packages. In addition, it was determined that there are issues with access rights on Linux machines. In order to circumvent the necessity of addressing these issues in the event of each new start script, it was deemed a prudent course of action to implement a more systematic approach. The solution contains all the necessary commands to successfully initiate it. Consequently, when the solution is implemented in a different system or modifications are being made, it is sufficient to execute only the start script to initiate all containers. The versioning of the entire system is performed in GitLab. In the context of Docker images, it is imperative to emphasize that only the data and configuration files, in conjunction with the delineated scripts and Docker (compose) files, necessitate versioning. This approach has been demonstrated to result in a substantial reduction of overhead expenditures.

6. Schluss

Bibliography

[87B,] § 87 BetrVG - Einzelnorm. https://www.gesetze-im-internet.de/betrvg/__87.html.

[Nod,] Node-red-contrib-influxdb. http://flows.nodered.org/node/node-red-contrib-influxdb.

[Que,] Query data | Get started with InfluxDB 3 Core | InfluxDB 3 Core Documentation. https://docs.influxdata.com/influxdb3/core/get-started/query/#Copyright.

[REG,] REGULATION (EU) 2016/ 679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL - of 27 April 2016 - on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/ 46/ EC (General Data Protection Regulation).

[Run,] Running Node-RED locally : Node-RED. https://nodered.org/docs/getting-started/local.

[UMH,] UMH installation requirements. https://umh.docs.umh.app/docs/getstarted/installation/.

[Wha,] What is IoT (Internet of Things)? | Definition from TechTarget. https://www.techtarget.com/iotagenda/definition/Internet-of-Things-IoT.

[Wri,] Write data to InfluxDB 3 Core | InfluxDB 3 Core Documentation. https://docs.influxdata.com/influxdb3/core/write-data/#line-protocol-elements.

[Nod, 2020] (2020). Node red minimun configuration - Hardware. https://discourse.nodered.org/t/node-red-minimun-configuration/31416/2.

[Alauddin, 2018] Alauddin, M. (2018). Process improvement in sewing section of a garments factory a case study.

[Asemani et al., 2019] Asemani, M., Abdollahei, F., and Jabbari, F. (2019). Understanding IoT Platforms : Towards a comprehensive definition and main characteristic description. In *2019 5th International Conference on Web Research (ICWR)*, pages 172–177.

[Ashton,] Ashton, K. That 'Internet of Things' Thing.

[Gustin and Jasperneite, 2022] Gustin, D. and Jasperneite, J. (2022). IoT Device Management Based on Open Source Platforms - Requirements Analysis and Evaluation. In *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*, pages 1–4.

[Jung et al., 2020] Jung, W.-K., Park, Y.-C., Lee, J.-W., and Suh, E. S. (2020). Remote Sensing of Sewing Work Levels Using a Power Monitoring System. *Applied Sciences*, 10(9):3104.

[Kang et al., 2016] Kang, N., Zhao, C., Li, J., and Horst, J. A. (2016). A Hierarchical structure of key performance indicators for operation management and continuous improvement in production systems. *International journal of production research*, 54(21):6333–6350.

[Kiron, 2022] Kiron, M. I. (2022). KPI (Key Performance Indicator) in Garment Industry.

[Ltd,] Ltd, R. P. Buy a Raspberry Pi 2 Model B. https://www.raspberrypi.com/products/raspberry-pi-2-model-b/.

[Muchiri and Pintelon, 2008] Muchiri, P. and Pintelon, L. (2008). Performance measurement using overall equipment effectiveness (OEE): Literature review and practical application discussion. *International Journal of Production Research*, 46(13):3517–3535.

[Neely et al., 1995] Neely, A., Gregory, M., and Platts, K. (1995). Performance measurement system design: A literature review andresearch agenda. *International Journal of Operations & Production Management*, 15(4):80–116.

[Quoc et al., 2025] Quoc, H. D., Thuy, T. T. T., Sinh, Q. N., Danh, H. B., and Lan, A. V. T. (2025). Applying IoT for Operations Management and Production Planning in Industrial Sewing. In Nghia, P. T., Thai, V. D., Thuy, N. T., Huynh, V.-N., and Van Huan, N., editors, *Advances in Information and Communication Technology*, pages 1054–1062, Cham. Springer Nature Switzerland.

[Rani, 2025] Rani, S. (2025). Tools and techniques for real-time data processing: A review. *International Journal of Science and Research Archive*, 14(1):1872–1881.

[Rose et al.,] Rose, K., Eldridge, S., and Chapin, L. The Internet of Things: An Overview.

[Stražinskas, 2025] Stražinskas, P. (2025). *Development of a Solution for Monitoring and Recording the State of a Sewing Machine /*. PhD thesis, Kauno technologijos universitetas.

[Tambare et al., 2022] Tambare, P., Meshram, C., Lee, C.-C., Ramteke, R. J., and Imoize, A. L. (2022). Performance Measurement System and Quality Management in Data-Driven Industry 4.0: A Review. *Sensors*, 22(1):224.

[thingsboard,] thingsboard. Installing ThingsBoard CE on Ubuntu Server. https://thingsboard.io/docs/user-guide/install/ubuntu/.

[Turki et al., 2024] Turki, M., Boussaidi, G. E., Benzarti, I., and Mili, H. (2024). Evaluating Open Source IoT Platforms: A GitHub Analysis. In *2024 IEEE/ACM 6th International Workshop on Software Engineering Research & Practices for the IoT (SERP4IoT)*, pages 14–21.

[Türkoğlu et al., 2024] Türkoğlu, A., Ersen, O., Yiğit, İ. O., Unal, D., and Golcuk, H. (2024). Comparison Between Time Series and Relational Databases. In *2024 9th International Conference on Computer Science and Engineering (UBMK)*, pages 1174–1177.

[Wedanage and Thiwanka, 2022] Wedanage, D. K. H. W. and Thiwanka, G. M. (2022). Fog Assisted Industrial IoT Module for Cycle Time Capturing in Apparel Industry. In *2022 International Mobile and Embedded Technology Conference (MECON)*, pages 352–356.

[Young, 2022] Young, D. T.-T. (2022). The Industrial Internet Reference Architecture.

Abkürzungsverzeichnis

IoT	Internet of Things
$IIoT$	Industrial Internet of Things
KPI	Key Performance Indicator
PLC	Programmable Logic Controller
$OPCUA$	Open Process Control Unified Architecture
$DBMS$	Database Management System
$TS-DBMS$	Timeseries Database Management System
$IIRA$	Industrial Internet Reference Architecture
V	Volt
x	x

List of Figures

List of Tables

A. Quellcode

1. Source 1
2. Source 2

B. Rohdatenvisualisierungen

1. Graustufen
2. Verteilungen