www.ijecs.in
International Journal Of Engineering And Computer Science
Volume 9 Issue 12 December 2020, Page No. 25289-25303
ISSN: 2319-7242 DOI: 10.18535/ijecs/v9i12.4587

Data Engineering Architectures for Real-Time Quality Monitoring in Paint Production Lines

Raviteja Meda

Lead Incentive Compensation Developer,
ORCID ID: 0009-0009-1578-3865

Abstract
A wide range of industries, including automotive, electronics, consumer goods, and food, relies on manufacturing processes to produce different kinds of products. Quality monitoring is critical to guaranteeing the quality of the products on the production lines. Various approaches have been developed to monitor the quality of the produced items on the production lines, including model-based approaches and data-driven approaches. Most of these methods only assert the quality of the items after the entire production process is finished using the resulting inspection results. Each produced item after passing the control quality monitoring step is either labeled “approved” or “not approved”, and the overall quality of the production process is asserted based on the quantity of the approval inspection results. This means that there is a significant amount of faulty items produced before being rejected. It is highly desired to conduct an automated real-time quality monitoring approach to alert the fault in the manufacturing process as soon as possible. Although a wide range of quality monitoring approaches have been proposed and developed successfully, they do not take into account the streaming nature of the data, and they have not been evaluated in streaming environments, where data is mined in the order they are generated and the training process is commonly never-ending. This article proposes the first work on real-time quality monitoring methodology developed from deep learning architecture known as Neural Networks with Dynamically Evolved Capacity (NADINE), namely NADINE. It is the extension of NADINE featuring 1-D and 2-D convolutional layers that process the time-series and visual data streams captured from the sensors and cameras of the production line, respectively. It is stated that NADINE is able to conduct an online quality monitoring task for both streaming time-series and streaming visual data, adaptively evolving its capacity by adding the neurons and their connection weights on the fly, and conducts training with the first-in-first-out replay buffer to gradually train the newly added capacities of the model.

Keywords: Data Engineering, Real-Time Monitoring, Quality Control, Paint Production, Manufacturing Analytics, Industrial IoT, Data Architectures, Process Optimization, Edge Computing, Smart Manufacturing

1. Introduction

Today’s paint production lines utilize digital technology that allows them to create precise recipes of paint products. A complex pump and valve system mixes the paint ingredients according to the specified recipe and requested amounts. Paint products usually consist of base ingredients and additives that need to be mixed and processed effectively to obtain the desired paint quality. Paint products are also characterized based on their specifications. For the quality monitoring of paint production lines, experts examine quality-monitoring features, such as the viscosity of the paint products, the percentages of colorants, etc., using measurement instruments. Although analytical laboratory instruments are accurate and less prone to error, the measuring process is costly and slow, giving rise to the demand for real-time automated paint quality monitoring using data-driven approaches.

Related to the important issues, such conductivity, or viscosity, of the paints over time within the production lines are still neglected, and hence, paint quality monitoring has to be conducted offline since the availability and quality of data in a data stream are the factors. To cope with these problems, a monitoring framework called DEAM is proposed, featuring target-oriented models that employ deep learning algorithms from a data-driven perspective to deliver the trade-off between the modeling performance and the monitoring objective.

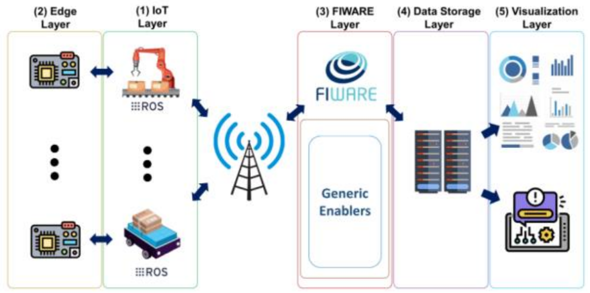

Fig 1: Real-Time Fault Detection and Condition Monitoring

1.1.Background and Significance

With the changes in consumer demands and preferences, the manufacturing landscape has undergone tremendous evolution which offers a range of customizability options in form and function to products. This leads to the advent of high-mix low-volume manufacturing which is a category of smart manufacturing. In the current era, HMLVM poses challenges to production as it demands a paradigm shift from the conventional mass-production line for either low or high-volume rigid systems to new and improved technologies. From the standpoint of production and/or supply chains, there are concerns on the decision to make-a-sort and its associated quality aspects. Specifically in paint manufacturing, production line optimization, time-event monitoring and quality monitoring are sought in view of new technologies and decision-support analysis. Current practices in paint production lines already exist but are of non-informative type.

Manufacturing monitoring has emerged as a research field of wide appeal featuring a variety of engineering applications including fabrication, assembly, painting and packaging. Current monitoring implementations have exploited to a fair extent sensor technologies and statistical process monitoring techniques, notably multivariate control charts. The superiority of these human-in-the-loop systems lies in their ability to monitor faults potentially undetectable by equipment onboard the machines.

Equ 1: Latency Model for Cross-Cloud Communication (L)

$$L = L_{intra} + L_{inter} + L_{proc}$$

Where:

- L_{intra}: Intra-cloud latency
- L_{inter}: Inter-cloud latency
- L_{proc}: Processing delay at endpoints

2. Data Storage Solutions

There are systems, based on the open-source DataFlow framework, which allow cloud-based real-time monitoring of the performance of the machines used in industrial production. These systems were presented, capable of adapting to the computational resources of the cloud, and implementing different machine learning algorithms to detect normal operation instead of anomalies in real-time on different types of raw data. In addition to these new systems designed to be used in production plants, two cloud processing failures have also been described. This information reporting is valuable, especially when monitoring critical processes. Different types of cloud-based data analysis systems will always analyze the data produced by the industrial machines, and alarms will be issued for abnormal behaviors. In order to ensure the correct implementation of the algorithms used to analyze the data on the cloud, it is necessary to accurately simulate the inputs to the systems. To this end, the simplifications considered when modeling and analyzing a paint production line have been described in detail. Additionally, algorithms used to recreate realistic data patterns and anomalies have been proposed.

The cloud-based data analysis systems are subject to failures, and crashes in either the producer or consumer node lead to the loss of data. Failure processes, which are common to distributed processing systems, have been explained along with their optimization. The analysis of the production process, from the moment raw materials enter any production line until they are characterized and packaged, is vital for compliance with market specifications and regulations. One of the installation’s purposes is, as much as necessary, to monitor the functioning of the production machine, both for triggers that alert the operator to anomalous cases and automatically record every step of the production.

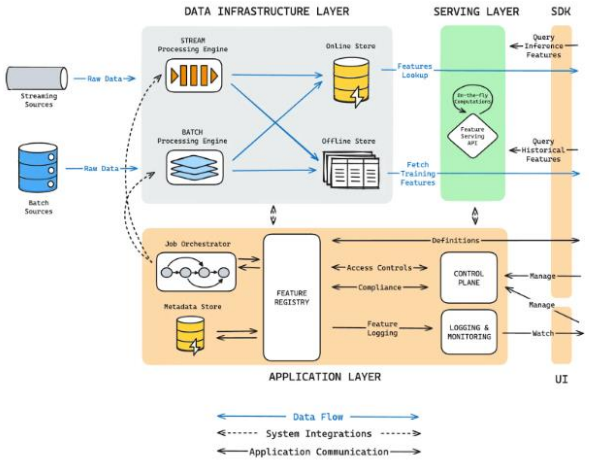

Fig 2: Data Storage Architecture

2.1. Relational Databases

Data from various sources must be gathered and combined before a complete report can be produced by issuing a set of real-time queries. This information is crucial for the management personnel to run the operation and to improve the cycle time in the FAB. MES has a complex architecture with multiple modules and databases with large sizes, making it challenging to respond to queries made by MES users. These queries produce real-time monitoring reports for management and customers, consisting of information about WIP status and history movements. Database performance monitoring helps DBAs and application developers to locate and identify the causes of bottlenecks due to poor database performance. Identifying and fixing SQL performance can help solve the performance problem. Therefore, there is a need to fix problematic SQL queries by understanding the problem and re-writing them. Relational databases are the most common and widely used databases today. There exists a fixed schema that holds tables with various attributes that follow the schema definition. There exist different types of relations, which are 1:1, 1:N, M:N, and self-join. Tables can be joined over common attributes. The main strength of relational databases is their scalability. Many open-source and commercial RDBMSs exist in the market, and they are the most common dwelling for enterprise applications. Most companies start with a professional RDBMS, and industrial products like ERP databases, MIS databases, etc., are maintained by them. Types of data that are well suited for relational representation are consistency critical, expensive to collect frequency queries, and with unknown long-term preprocessing costs.

2.2. NoSQL Databases

Due to the rapid growth in the size of data going into modification and analysis, it is being realized that massive technological adjustments and extensible processing resources are required by the current database systems to cope with large data volumes. This implies that new systems need to be considered to keep support for desired performance guarantees on such large volumes. Consequently, firms expect many business benefits by investing in larger-scale data analytics both immediately and in the long term. Implementing that relies on a given system or model for streamed data processing that is best for employing multi-sourced, variable rate data filtering and acting promptly on modifications. For the automated monitoring systems, there is a need to implement sufficient accuracy and meet given time countermeasures to detect failures promptly.

Different NoSQL platforms provide a solid language for data storage, query, and statistical analysis. One platform shows an expected layout of how data will start again and tables relations within the datasets. It is actively built on servers responding to queries, for example, finding records built on a given condition of a field constraint, returning such documents. The data models respond to modifying structural revenues with expanding elements and documents. It contains internal functions, basic sequence operators for pattern count and document containment, and indexes. The raw data is first pre-processed into a proper format for database insertion in standard format files containing target field and document structure. For a number of observations, season start time is built on monetary and time granularity levels, and then data is stored in collations. It handles two types of collection to quickly build, handle, and analyze complex data given a defined model of expected growth. The platform was implemented with a comfortable experience of task execution at the data source level. The quality check system architecture was proposed generally.

The platform was defined for development which has a shell to quickly check the database status and analyze stored data similar to other major languages. It should be noted that a visualization tool can be plugged in with the platform to connect and provide real-time dashboarding functionality. The NoSQL platform was confirmed as a flexible solution for any big data, running with large scale, simple architecture, and low maintenance costs. The look-up tables must be prepared manually initially. But this is a one-time job that does not require frequent changes.

2.3. Time-Series Databases

A time-series database (TSDB) is a dedicated database that is optimized for the storage and retrieval of time series data. TSDBs inherently provide streaming capabilities, but it is not always guaranteed that they are the best fit for monitoring systems that ingest high-velocity data streams. They generally enable effective use of compression and encoding mechanisms applied to time-series data, while introducing optimizations for aggregate queries. Recently, many new TSDBs powered by new distributed technologies have been proposed, and correspondingly, new methods for time-series monitoring that leverage these databases. TSDBs scale fault-tolerantly in a distributed architecture to very large amounts of data and provide batch, ad-hoc, and continuous queries for analyzing this data.

Most existing time-series databases are derived from general-purpose databases and use relational table structures augmented by customized indexing and query processing. Also, some commercial cloud databases provide a cloud-native service for time-series data which is backed by proprietary internal architectures similar to columnar distribution. Unlike traditional databases, PolyFS writes incoming data into immutable appended files that are later rotated to a read-optimized format. It works well for storing time-series data which is written in a relatively small number of bulk load operations.

The need for having very low cost per query led to a design that stores data as (1) sorted data files for raw data, (2) aggregate data files for metrics, and (3) bloom filters for fast predicate filtering. Targeting on sub-second queries, CQ2 design leverages existing OLAP technologies. CQ2 builds queries in user-defined SQL completions and re-schedules them for low latencies. The query output is further pipelined to analytic workloads. CQ2 also supports a sophisticated monitoring system that ensures high availability of the entire work scheduling pipeline. It utilizes both system metrics from cluster managers and user-defined metrics from query scheduling systems to construct a statistical-based monitoring pipeline.

3. Data Quality Assurance Methods

Quality monitoring is a crucial task in manufacturing that aims to find out whether a produced item conforms to the desired specifications or not. The common practice of quality monitoring in production lines relies on manual inspection performed by human operators known to be slow and prone to scanning error or missing defects. This issue raises another interest in research and industry for automated real-time quality monitoring. In the data-driven approach to automatic monitoring, three components are needed: a quality definition, a database that records the quality-representative attributes, and a monitoring model that will tell whether the item is good or defective. However, existing attempts for data-driven approaches are not directly applicable to quality monitoring tasks in modern industries since the raw material quality information is captured from streaming data in an online manner. Therefore, the data collection phase should be carefully considered in order to produce a better monitoring model. On the other hand, the prevalent data-driven approaches utilize off-the-shelf machine learning algorithms which are more sensitive to subject engineering problems and rely heavily on hand-crafted features for defining the injured state.

The proposed online quality monitoring methodology is developed from a newly introduced architecture of the deep learning algorithm called NADINE. Realizing that sensory information in terms of time-series signals is in a streaming manner, continuous representation of time-series information in the form of a data guide representation and a reconstruction model called the encoder-decoder architecture is proposed. The data guide representation models the time-series signals into a smaller-size sequence of time-frequency representations that preserves the integrity of the information of interest, making it directly useful for various monitoring tasks. Upon rigorous analyses, it is found that since the audio signal is basically in a streaming nature with large temporal changes, past observations of time-series data would still be useful for the prediction target. Therefore, modeling the data guide representation into a sliding window approach with an appropriate hyperparameter tunings of input window and searching horizon length is proposed. There are two main tasks of data quality assurance in this system. First, in order to find a copy of the inspection criteria, three traditional feature extraction methods for each achievable data type are applied. The monitoring model is built using a machine learning algorithm, which is then used in routine batch-mode monitoring of the produced items. The output of the monitoring model would return an alert for quality assurance personnel for further examination. Second, real-time monitoring requires fast data representation and model execution. In order to assure that the data representation delay does not hinder any defected item, a consecutive representation of the time-series signals is required and is also validated on the process dynamics disturbance occurrence. The efficiency of the online quality monitoring approach is further demonstrated by conducting a series of real-time experiments.

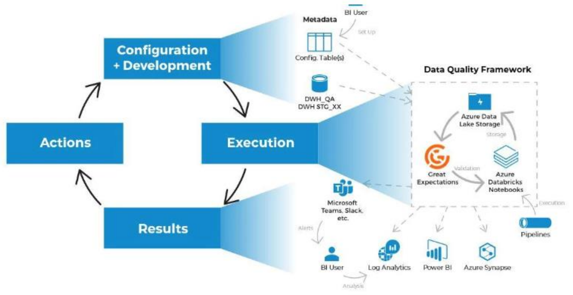

Fig 3: Data Quality Framework

3.1. Data Validation Techniques

Despite the different levels of experience in these techniques, data preprocessing is one of the aspects that impacts the final generalization performance of the models. Starting from the raw data of the production and after a review of the techniques used by the entire scientific community, analysis and selection are made regarding their applicability to the problem. The first step is data cleaning, grouping and formatting the data in a new dataframe. Deep learning architectures require well-structured data and one of the biggest drawbacks of industrial data is that it is usually scattered across several databases in different formats and levels of detail. Often, data validation errors occur when methodologies and automated processes are not implemented to monitor the data quality. The most common data quality issues are spelling errors (case sensitivity), date format differences, missing values, and outliers. Therefore, a data validation process in terms of the above-mentioned issues is implemented. This first implementation, although it does not solve the problem of decision gaps introduced by the manual inscriptions, aims to evaluate how much the latter alone affects the generalization ability of the algorithm. The second part of this section presents the data preparation techniques known in the literature proposed for time series problems. The techniques are originally created for machine learning architectures but can be easily adapted to deep learning models. The data preparation techniques considered for the analysis are: * Resampling frequency, which creates a new index with the defined frequency and adds the necessary data points from the original data frame. * Correlation-based reduction, where the maximum lags of each variable are defined with the maximum lags being added to the original data set. Also, some new variables can be generated based on correlation with output variables. The proposed above-mentioned data preparation techniques do not overcome the decision gaps as they mitigate the risk of a less accurate model due to data overfitting. However, it is important to evaluate how much they affect the generalization ability of the model with these settings.

3.2. Anomaly Detection

Anomaly detection finds irregularities different from the expected normal behavior in data. Anomaly detection can be an unsupervised, semi-supervised, or supervised machine learning task, which basically implements different approaches to the problem. Each of these approaches has its strengths and weaknesses that can help drive the solution approach and enable one to choose the most appropriate method based on the list or maximum number of possible deviations. While unsupervised approaches attempt to detect outliers in datasets without any prior knowledge of the domain, semi-supervised methods require training data describing only the normal behavior, and supervised methods use labeled data points in both classes. Depending on the scenario, the process might either require only the training of normal behavior models, for example, in the absence of outliers reaching the current state of the processes a few weeks after product ramp-up.

The approach consists of matching real-time and lagging data columns during momentary and on-demand data collection processes. Once matched, pre-processing steps will take place on these raw data points to filter each data frame according to the specific process chain of best paint or varnish usage. The data will then be transformed into the input format of the selected anomaly detection algorithm. A recent combination of graph-based and classic learning approaches will be configured for anomaly detection.

Equ 2: Automation Coverage Ratio (ACR)

$$ACR = \frac{N_{auto}}{N_{total}} \times 100\%$$

Where:

- N_{auto}: Number of automated tasks/components
- N_{total}: Total number of tasks/components

4. Data Sources in Paint Production

In the paint production industry, it is essential to capture and dynamically monitor a variety of information throughout the entire paint production life-cycle in their production lines. This information comprises all quality-relevant production parameters, process data, and tangible characteristics of the raw materials, intermediate products, and final goods such as chemical and physical properties, temperature, pressure, density, and viscosity. A variety of inexpensive and widely available sensors and measuring devices are employed by the paint production industry to record this wealth of information, which is processed periodically or cumulated at certain milestones. The continuous digitalization and industrial internet of things (IIoT) enhance the possibilities of implementing next-generation whole-paint data value monitoring. It enables the possibility of streaming a variety of information from each individual node of a production line while including further data integration. This establishes novel ways of looking at production and quality data streams, enabling a different quality mindset. However, RCS implemented in paint production lines primarily caters to discrete events at a wider granularity. The majority of the quality monitoring is aggregation-based, resulting in the loss of relevant information in an 'information cascade' manner. As a consequence, valuable quality data streams may not be further processed with online methods, missing noticeable opportunities of quality assurance for flexible production scenarios [6]. The recent developments in paint production lines require a shift to dynamic parallel data engineering architectures, which are capable of real-time processing and rendering data scalability. Real-time data monitoring and modeling on all quality-relevant information provide invaluable information for the entire lifecycle of production data, but result in enormous complexity and costs extending beyond the paint production domain. Streaming quality monitoring requires stream processing to provision on-demand quality monitoring on single quality-relevant data, while distributed global data pipelining renders data engineering architectures capable of hierarchical and online analysis on quality monitoring across each independent analytics. In response, this explores data engineering architectures for the realm of real-time quality monitoring of paint production lines leveraging the whole data continuum, namely on-frame-stream data integration. The main contributions include the first draft of a data engineering architecture proposition for painting quality monitoring, composition patterns of entire data monitoring processing models, and a corresponding data pipeline engine.As an evaluation use case, data from a Uhlmann B1440 packaging machine is analyzed. The packaging of the products is triggered by a camera detection of the confirmative presence. Data of 10 sensors are recorded at a resolution of 1 ms over a period of 44,101 s with a sample rate of 1000, resulting in a total data volume of 2566 MB. The monitoring task, which is often the main concern in data analysis and the first step of their evaluation, is to detect significant deviation from the established normal behavior of a system. The proposed solution consists of a signal-driven processing pipeline to derive various features and indicators from a variety of sensor data and a data-driven monitoring solution, both of which are valuable and generalizable for many other applications.

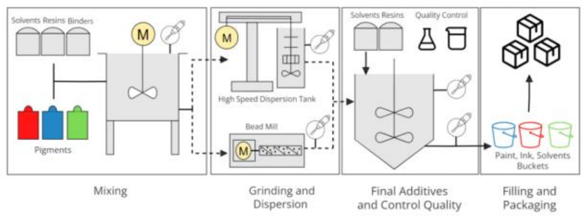

Fig 4: Data Sources in Paint Production

4.1. Raw Material Quality Data

In the paint production company, different batches of raw material are received from different suppliers, and each batch is analyzed for its physical and chemical properties to assess its quality. Table 1 shows the list of tests performed on each batch of raw material. All the test results are maintained in a SQL Server relational database. There are two main reasons why only physical and chemical parameters are chosen for quality assessments: (1) The UV-transmitter tests are not conducted in all cases; instead, either a salt spray test or MTTF tests are performed, depending on the type of resin, and (2) Some rainfall tests are not applicable to certain products or batches of resin. However, necessary alarms and work instructions have been developed based on the Moisture, Salt spray, MTTF, and rainfall test results, and their frequency is illustrated in Figure 14 for different categories of finished paint (Alkyd, PUD-acrylic, etc.). In addition to these physical and chemical properties, there are two additional parameters in the database: Supplier code (abbreviation of supplier names) and Yellow ID (a number assigned to raw materials of a specific supplier regarding price negotiations). The absorption of dry pigments in paint is a measure of the mass of pigment that is bound to dry paint. It is an important attribute of paint which affects the hiding power of paint. Moreover, due to different supplier compositions of resin types, density and viscosity of paint brands are not explicitly specified by the paint producer. Instead, the paint producer specifies the range of values for these attributes, and for most of these parameters, upper and lower specification limits are found. Whether a quality assessment is performed for a raw material attribute or not is also recorded in the quality analysis of raw materials database (for example, for paint with production line code 006, the specific attribute is not tested and the assessment is not needed).

4.2. Process Parameter Data

With the introduction of Industry 4.0, sensor technologies have been widely used in a variety of business processes and applications, including smart factories. While the ongoing trend results in the generation of more and more sensor data, an effective solution for the processing and analysis of sensor data has not been commonly found in real-world use cases. Particularly in sensor data analysis, the currently widely adopted data-driven solution does not provide a systematic way to acquire the knowledge, depending on the available raw data, and is unable to detect unknown process anomalies that have not been trained beforehand. To tackle the above problems, it is proposed to apply a signal-driven solution to build a system for the processing and analyzing of sensor data. The framework, consisting of a signal-driven processing pipeline, should systematically exploit expert knowledge on the physical processes to derive a variety of features and indicators, including fixed and time-evolving ones. It is also proposed to apply a recent technique of a data-driven multiscale anomaly detection to analyze the derived fixed features and time-evolving indicators for a holistic overview over the analyzed process. The signal-driven holistic monitoring (SHiM) system is demonstrated with a real-world application in a packaging machine. A complete framework consisting of a processing pipeline and a monitoring solution is established to analyze industrial sensor data.

4.3. Environmental Conditions Data

For many manufacturing lines today, environmental factors play a critical role in the final product's quality. Quality issues caused by out of specification environmental conditions are not discovered until the product is near the shipping line. Sometimes this could be several weeks after the product was manufactured. Catching defects earlier in the process would mean that the cost of rework or scrapping a product would be far less than if the product was almost ready to ship. Manufacturing lines will have a centralized measurement for environmental factors, usually temperature and humidity. Some lines may contain multiple sensors throughout the room to control environmental zones. Rarely does the data linked to manufacturing systems allow engineering to correlate environmental conditions to specific products in the line. This paper describes a low-cost and flexible wireless environmental monitoring solution that does just that. In this system, each product in the manufacturing line is affixed to a sensor pack which is capable of measuring temperature, humidity, and motion. The sensor pack wirelessly transmits its data to a microcontroller which relays the information to a cloud service that handles real-time storage, processing, visualization, and alerting. Completed environmental maps can then be overlaid with the production system to correlate quality defects with the environment products were manufactured in. As manufacturing automation and data gathering technology are improving, many savvy manufacturers are leveraging existing data to gain more insight into the manufacturing process and to drive quality improvements. Manufacturing processes produce huge amounts of data typical of time series data such as temperatures, pressures, motor currents, and encoder counts at high rates from various sensors and loggers. Although manufacturers are using standard SQL databases for storing data, such databases cannot deliver a satisfactory user experience when it comes to analyzing and visualizing time series data. On the other hand, there are already several highly scalable distributed solutions for time series data. In this paper, a distributed pipeline for time series data transportation, storage, and visualization is presented that attends fast and scalable exploration and analysis of manufacturing quality data.

5. Real-Time Data Processing Frameworks

Real-time processing frameworks are designed to process large volumes of data coming from different sources and to generate data in real time. They also provide a distributed, scalable, fault-tolerant, and reliable architecture. Many frameworks are ready to be used. Some can be combined transparently. Among these frameworks, some are dedicated to streaming; others are designed for batch processing but can be adapted to streams with limitations. However, there is no classification or descriptions of which frameworks can fit a certain need best. Therefore, this work presents a description of various frameworks, with an analysis of their ability depending on the chosen design aspects. This framework description is divided into two categories based on whether they are dedicated or hybrid. It provides a basis for designers and researchers needing to choose a framework for data processing. Suitable streaming frameworks have been described based on aspects such as scalability, simplicity, installation, literary coverage, pricing, supported languages, and readiness for acceptance from our work, which have been commonly neglected.

Many data stream processing frameworks have been designed and implemented. Each of them comes with a different processing model and approach, and they differ in their target applications and purposes. The implementation is often incomplete compared to the original prototype. Little information about the finer aspects of these models is available even in scientific literature. Some frameworks are dedicated to streaming, while others are designed for batch processing but can be adapted to streams with some limitations. A few overviews of the existing stream processing frameworks are available. The first is more like a survey, capturing more than 50 existing frameworks, but this work only describes the coarse-grain features of the frameworks such as an overview, implementation, and a few key aspects, which are not insufficient in evaluating and selecting a suitable framework to suit certain needs. The second is an extensive comparison of the batch processing frameworks, which evaluate each of these frameworks' performance over aspects such as efficiency and scalability rather than designing them based on the application needs.

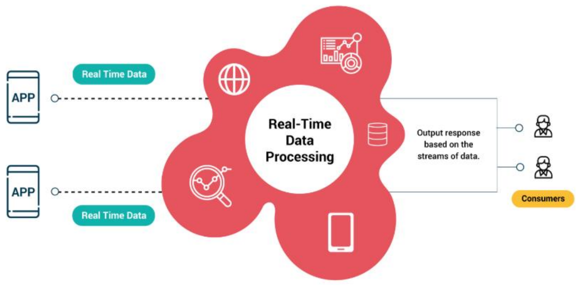
APP
Real Time Data
APP
Real Time Data
Real-Time Data Processing
Output response based on the streams of data.
Consumers

Fig 5: Real-Time Data Processing Architecture

5.1. Stream Processing vs Batch Processing

In this regard, the fundamental distinction between batch processing and stream processing systems needs to be discussed. For both systems the same categories of functionality apply for storage, queries and analysis, but their control of time is different. As the name suggests in batch processing, time is taken into account in terms of batches also called windows and batch periods. In terms of an operation in this system, the time period is taken and the data arriving in that time period are processed. This could be done in every fixed interval of time or when just available data are needed for this operation. In any case, there is a waiting time for processing and results are not immediate but delayed. Stream processing systems on the other hand use the term stream when referring to data or input and result. Time is taken into account in terms of its unfolding or continuously and instantaneously. Data arriving at the systems as it unfolds are processed as soon as possible in analysis or queries. Under such conditions of time, analysis cannot be done over all data but for a few previous data in the case of continuous queries. The result also is not a point and deltas or an interval output is generated. Display of one particular data or its changes is usually expected. In every case the analysis output follows the data arrival as soon as possible or near real-time and without any waiting time. In general employment of the two processing systems together is not uncommon as the research cases under consideration understand how both systems can be integrated. As just discussed traditional processing systems are batch processing systems and the advent of massive and ever growing data has prompted the need for better data analysis tools that could handle process data in more real-time fashion. The emergence of stream processing systems has been spurred from this need. As for the type of processing, both stream processing and batch processing systems can be grouped based on the nature of processing employed. Basic operations of join, filter, windowing and aggregation are common for both systems. Batch processing systems on the other hand for instance use MapReduce style merging, partitioning and column storage, while detecting changes and querying in common are employed for stream processing systems different types of spatial, temporal and relative change detection families have been designed.

5.2. Popular Stream Processing Tools

With the rapid developments in technology both in data processing and production systems, data generation is now possible at an unimaginable scale. This has created new opportunities for organizations, but also, due to the velocity and batch-less nature of data generated, new challenges have arisen. Streaming platforms allow the ingesting, processing, and storing of data, at many terabytes a day or more, while the data is still relevant. Event- and stream-based systems provide solutions to real-time business processes by allowing for scalable and reliable stream processing, which is the basis for the architecture proposed in this paper. This chapter first gives an overview of the most popular open-source tools. These dark-horse technologies with a large user-base evolve rapidly, making them ideal for new projects. Batch processing tools are often included, as they are complementary to streaming processing.

Apache Kafka is a distributed messaging-oriented middleware, often wrongly framed only as a publish-subscribe message broker. It acts as a highly reliable distributed database, and middleware that can work with multiple producers and consumers, even many at once. The components of this ecosystem are: Brokers: Kafka brokers handle incoming streams of messages and store these in durable logs. Scalable clusters are also possible. Each message is stored as an immutable record with a unique offset for identification. Partitions: Each Kafka topic, that is a named output stream, is divided into multiple ordered logs, called partitions. This allows messages to be spread over brokers, only limiting the total load a single broker has. The spreading of partitions is handled entirely by Kafka itself, which also replicates the partitions across various brokers to make the stream resilient to broker restarts. Producers: Producers are data sources able to send messages to Kafka topics. Consumers: Consumers are data sinks able to read messages from Kafka topics. This can consist of external systems or stream processors. A consumer group consists of one or more consumers, allowing for consumer-load balancing.

Similar to SQLite for batch processing, Apache Derby is a distributed event-based in-memory database good for prototyping stream processing engines due to its compact size and bundled dependencies. It can, however, only store small log files and therefore is not feasible for production systems. Apache Flink is based on GroupBy and reading time windows, which will always produce semi-results instead of instant results. The process of reading a stream of messages is done using sources, which can be simple files, clients connected to remote queues, or other stream processors. Flink is robust and fault-tolerant, but not implemented in Java, making it harder to use.

Equ 3: Cloud Resource Utilization Efficiency (RUE)

$$RUE = \frac{U_{actual}}{U_{provisioned}} \times 100\%$$

Where:

- U_{actual}: Actual usage (CPU, memory, etc.)
- $U_{provisioned}$: Provisioned capacity

6. Architectural Design Considerations

A combination of MQTT protocols for data publishing and ExtraHop for real-time monitoring is needed for a viable technology stack. Measurement devices (temperature, humidity, and pressure) should have a data publishing frequency of 1 Hz, while image acquisition devices (visible and near-infrared spectrometers) and more complex measurement devices, such as the SIL Nozzle, should exhibit a higher frequency of 10 Hz, as they provide data more quickly than the other devices. For all devices, it should be possible to define and configure triggers (threshold conditions) for sending alerts regarding pending thresholds and send these alerts through MQTT protocols as soon as the condition is met. These alerts should also be sent through a Twilio phone message, which would be a good user experience implementation. ExtraHop is a real-time web-based monitoring technology that could be acquired for a price of 2790 euros/month.

A two-device architecture for the edge node is required: a cloud proxy for publishing all the devices' streams at a frequency of 1 Hz, and an on-site process that executes all Python scripts for publishing streams with a frequency of 10 Hz. Regarding the processing devices, Apache Airflow, which is relatively inexpensive and easy to use, will execute all the data transformation scripts based on the streaming published by the edge node. For visualization tools, Power BI should generate dashboards with all the visualizations and monitoring tools written in Python. It is also possible to integrate them with the PHP web application that controls the manufacturing process and its data warehouse to develop more complex dashboards.

Regarding the decision-making system, it should be integrated with Apache NiFi, which has an intuitive graphical interface for defining the workflow for processing the health status of the streams. The user interface for configuring the thresholds for alerts can also be built in this graphic tool. However, it only allows configuration to be done from the site where it is installed, and this control should be remote. Apache NiFi has a REST API to enable user interfaces to configure it from anywhere.

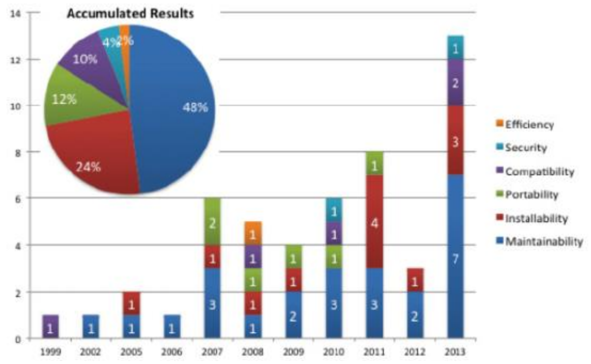

Fig : Bar chart representing the quality characteristic concern

6.1. Scalability

In this section a description of the scalability strategy is presented. Both possible configurations that will allow to achieve the expected scalability levels for the proposed Data Engineering Architecture will be defined. Currently, only the standard installation options are considered, in a future version of this document, possible hybrid cloud architectures will be considered. The integration of monitoring strategies is implied in the system's design of these architectures.

Given the requirements defined in the monitoring case study, scalability needs to be distributed horizontally. All the components of the Data Engineering Architecture can, in principle, be distributed on different nodes or containers. For this reason, tasks are divided into different units that can be executed independently in different threads. Scalability needs to be addressed in a “best effort” way as the production process is monitored in several steps. Each step has different scalability requirements. It is essential, for instance, to guarantee a very large number of records processed per minute in the collection step regardless of any other consideration, such as cost. A very large volume of data written per minute in the Buffering and Processing Steps must also be addressed, though some loss of information may be tolerated. Ingestion buffering step needs a high degree of parallelism to ensure a streaming pattern is followed in the whole pipeline. Broadly speaking, each Data Engineering Architecture component must fulfill an SLA with a TPhoneModel period delay. For components that can be scaled horizontally, instances can be added. Alerts and alarms in the Monitoring System and level-elasticity of Processing Components in the Data Engineering Architecture can be defined to notify the Task Manager about high levels of CPU or memory, or backlogs.

Two operational upper bounds dimensions should be considered for scaling nodes hardware use: increase computing capacity of a single node, only reducing the number of components or their replication on single units; add nodes at the expense of extending the cloud services used. This last parameter addresses at most the approach of introducing stronger competitors components, or the scalability levels of the Infrastructure Architecture, which explicitly require a number of nodes in a range of 1. Where it makes sense, the expected costs of the upper bounds for the conditions previously stated must also be presented. It is important to include whether or not there is a whole step or additional elements that would be required to meet defined bounds.

6.2. Fault Tolerance

Various architectures for temporal fault tolerance are proposed that rely merely on the existence of data stored in reliable storage. After presenting their proposed architecture and design methodology, they argue how the use of already available persistent storage for consistent storage (i.e., storage of processing state) and re-playable sources (i.e., storage of input data streams) for fault recovery should be treated as a new architecture and design by itself, and how it could allow both availability compared to existing systems and facilitating an increased richness of prospective applications. In the context of the novel architecture and design, temporal fault tolerance will be explained in terms of spatial fault tolerance with a main difference being the recovery mechanisms, and how the wealth of knowledge known for both can be brought together.

The ongoing trend towards sensorization, machine control and increased monitoring in all but the simplest applications lead to an observation of audio and visual data needing to be processed as it comes in through distributed acquisition, embedded onboard advanced streaming processing, and immediate accessibility through streaming delivery with a very high level of temporal fault tolerance. Multimedia broadcasting and personalized video on demand are among commercial applications of systems already implemented in parallel platforms. However, auto-scoring parameters and the resulting raw measurement streams from commercial paint production lines have serial and hybrid video-on-demand architectures forcing long retrieval and preparation times. New architectural and design models are proposed that would allow a new augmented richness in architectures and designs previously only supported through spatial fault tolerance.

Systems have become a necessity for processing vast amounts of data with never seen before rates, format transformations, and data enrichment in running time. With continuous processing being currently the best effort service in terms of timeliness and availability, streaming fault tolerance would provide a new quality of service for tradeoffs between availability, timeliness, and consistency. New error models for systems have and are still being designed and populated with probabilities of different faults per component (functional and temporal) to which existing system architectures dealing with such errors have been reviewed. By leveraging the knowledge and implementations of systems, a new means of increasing the availability of systems by treating persistent (and re-playable) data storage as a computing node and applying a similar architecture view, error model, and tools for systems to improve their designed and scheduled processing and storage episodes are provided.

6.3. Latency Requirements

The standardization of quality control parameters occurs offline wherein the samples are compared against a baseline set of paints post curing process and categorized into acceptable and reject. The latency incurred to send the samples to the QC lab and obtain the results of the test is larger than the production cycle time. Each UV paint cycle takes approximately 1 min and on going for every batch of paint put in production. In contrast a full set of paint colors has more than 4000 test values which contains more than 400 images that incur around 5 minutes of processing time to build.

The ultra-high-performance inspection camera is proposed wherein it can transmit 36 images per second over 5G. This can be streamed live and monitored over cloud servers with 5G infrastructures.

Batch size industrial automation refers to a vastly different process, wherein orders are more variable, products in each run are not identical but almost identical, many jobs are completed in a short period of time that involves different types of materials, machines, and operators. If an online solution is being considered, one promising approach to consider is a relocalized non-linear forecast based on. In this setting, a small batch that needs to be repaired or a small period of imprecision can be modeled to analyze the latencies. Recent research has explored data quantization and AM methods that tax higher quantization for less informative pixels hence making inference faster, hence this will be a good avenue to explore the trade-off between accuracy and latency.

7. Data Ingestion Techniques

Advanced Data Ingestion Frameworks, Techniques and Architectures for Comprehensive Data Analysis in the Era of IoT, Big Data, and Streaming Processing from Distributed Sensor Networks, Wearables, and Industrial Machinery and Production Lines

While traditional data ingestion frameworks were mainly processing static data in ETL processes, with the advent of the internet of things, big data, and stream data sources, the entire range of data processing architectures and frameworks has dramatically changed. Nevertheless, industrial companies have collected huge volumes of sensor data and are currently searching for suitable data procurement strategies. Thus, the very first steps integrated inline data preprocessing methods into the architecture of a stream analytics framework for industrial machines and production lines. The newly suggested low-cost microservice-based approach was able to achieve real-time analysis on an adequately powerful architecture. However, there are still constraints which can prevent comprehensive data analysis on such streaming data sources. Large model parameters and reference intelligent heuristic knowledge currently cannot be integrated into the cloud. Further, configurations for hybrid architectures must be chained and parameterized with respect to the streamed data source. However, considering the recent developments in consumption of streaming data, and also considering the available know-how, algorithms, and user knowledge, it was deemed necessary to architect and build up a new approach which allows the integration of any kind of data analyzation strategy. Such analyzation schemes are considered to be ontologies, or maybe even plain knowledge, which can be stored in worldly knowledge bases and application ontologies.

The considered long-term vision takes into account such endeavors for a data-agnostic deployment framework for continuous data analysis of structured and unstructured large data sources, including a cloud deployment model for shops with heterogeneous distributed production lines and subsystems. The fundamental building blocks of the architecture and a streaming based framework for jargon agnostic integration of reusable data analysis strategies, based on the specification of ontologies, have been elaborated for sophisticated applications on the edge. Summarizing, the architecture on edge/cloud hybrid systems for highly distributed streaming data sources is extended to obtain a platform and conceptual framework which allows every user to add analyzation strategies regardless of programming skills or data source type. Streaming fluent API is introduced for a semantically-annotated definition of reusable data analyzation ontologies. Preprocessing and transformation tools for sensor data are provided, with a configuration on how to link to a data source, transform streamed data and analyze it. Data source models characterized by simple ontologies are offered to provide user-friendly hooks for access to streaming data sources.

7.1. Message Queues

The intelligent monitoring and supervision of industrial production lines using data engineering architectures is a need that has been addressed in recent years thanks to the rapid growth of IIoT and data computing technologies, cloud services, and machine learning. In this regard and due to the wide use of paint-coated products in many industries, the challenge of real-time quality monitoring of the paint application process in industrial production lines has been presented here. Intelligent software architectures that are able to ingest real-time manufacturing process and equipment status data listening to different protocols, to store them in different repositories, and to provide real-time alerting messages for the occurrence of outlier events compatible with IVeT standards have been set up and tested with real-life production data.

These data engineering architectures are composed of five interrelated components: one or more stainless steel tanks equipped with advanced sensors for process quality monitoring (hardware layer), a commercial historian database and two different cloud services for data storage and outlier event detection; design software for the creation of manufacture execution systems and cloud-distributed data processing and control architecture. This architecture eases the monitoring of paint production processes using data-driven and cloud-distributed technologies, which are robust to support environmental variations. Feasibility and efficiency of the proposed architecture for hosting smart monitoring applications, as well as its capability to foster a continuous process smartisation, are actively demonstrated in the life cycle of different use cases. The sophisticated yet user-friendly configuration tools of the architecture ease the development of complex machine learning applications, paving the way for future infinite additions of new data-driven and cloud-distributed applications .

Cloud computing has been one of the most disruptive technologies, emerging in the 21st century that offers new computing architectures through service providers. Thanks to its distributed architecture, cloud computing overcomes the limitations of traditional computing systems with respect to massive growths in data volume and complexity and growing resource and time demands for horizon scanning, data-mining, and decision-making applications. Cloud computing is reshaping the design and development of industrial manufacturing processes, helping them to build, host, call and manage complex and heavy simulations and processes until then only feasible under supercomputer architectures.

7.2. Data Lakes

Data Lakes: A Survey of Functions and Systems—Data Lakes have gained popularity in industry to store wide-ranging, heterogeneous data in raw formats. They are an appealing alternative to Data Warehouses and play a vital part in a company's analytic strategy. However, there are very few mature systems with integrated functionality. This survey goes beyond the well-researched concepts of

Data Lakes and investigates existing Data Lakes and similar systems with a focus on their functions. It presents observations about current operating systems and an analysis of their advantages, drawbacks and applicable domains. The result is a comparative framework that analyzes according to characteristics like basic functionality, oversight capabilities, close integration and machine learning suitability.

8. Conclusion

In this paper, we proposed a complete data engineering architecture for real-time monitoring of quality information from a paint production line. The architecture is constructed for handling streaming data from various data sources, applied to the case study of autonomous quality monitoring of paint splatter information on car bodies in painting production lines. User-driven quality metrics, data management and ETL design, and engineering design for operationalization of machine-learning models are the ground for the repository of architectural components that we presented along with its concepts and terminology. We also presented our work on deep learning models to extract paint splatter configurations from visual data and sample quality trajectories alongside current enhancement efforts to adapt these models to online settings. The proposed architecture is built on a holistic and modular approach to data engineering, serving as a foundation for various application areas, datasets, and streaming environments.

While our work is proof-of-concept for the applicability of streaming data engineering methodologies in monitoring paint quality metrics, there's a need for greater academic and technical detail if the research and code are to contribute effectively to the community. Important areas for ongoing work are: (1) Developing extensible libraries of architectural components, spanning ingest, anonymization functions, machine learning models and data product visualization techniques; (2) Providing a cloud-hosted proof-of-concept for deploying the full architecture in projects similar to the one presented in this paper; and (3) Augmenting the current work on deployment and productionising machine learning code with in-depth technical examples on how this was done.

8.1. Future Trends

With the rapidly growing demand for automated quality monitoring in paint industries, there are still many improvements possible in the proposed architecture namely E2D. The presented architecture is a static architecture and a new strategy could be developed for the dynamic architecture. Due to the nature of the evolutionary algorithms of the proposed IDEA and the incremental E2D, both of these strategies could be combined with the well-established deep learning population-based optimization protocols as a very interesting future research direction.

9. References

1. J. Fritzsch, J. Bogner, S. Wagner, and A. Zimmermann, "Microservices Migration in Industry: Intentions, Strategies, and Challenges," 2019.
2. S. Talja, "Service-Oriented Process Models in Telecommunication Business," 2013.
3. Routhu, K., Bodepudi, V., Jha, K. M., & Chinta, P. C. R. (2020). A Deep Learning Architectures for Enhancing Cyber Security Protocols in Big Data Integrated ERP Systems. Available at SSRN 5102662.
4. K. Maroukian and S. R. Gulliver, "Leading DevOps Practice and Principle Adoption," 2020.
5. L. Silva, M. Unterkalmsteiner, and K. Wnuk, "Monitoring and Maintenance of Telecommunication Systems: Challenges and Research Perspectives," 2020.