Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 93 (2020) 724–729

www.elsevier.com/locate/procedia

53rd CIRP Conference on Manufacturing Systems

A framework for designing data pipelines for manufacturing systems

Omogbai Oleghe[a]*, Konstantinos Salonitis[b]

[a]Rite Pak Co. Ltd, Lyson Chemicals Avenue, Ota, 112242, Nigeria
[b]Manufacturing Department, Cranfield University,Bedfordshire, MK43 0AL, England

* Corresponding author. Tel.: +234-8033008484; *E-mail address:* o.oleghe@ritepaklimited.com

Abstract

Data pipelines describe the path through which big data is transmitted, stored, processed and analyzed. Designing an appropriate data pipeline for a specific data driven manufacturing project can be challenging, whereas there is a paucity of frameworks to guide one in the design. In this research we develop a framework for designing data pipelines for manufacturing systems. The framework consists of a template for selecting key layers and components that make up big data pipelines in manufacturing systems. A use case is presented to provide an illustrative guideline for its application. Benefits of the framework and future directions are discussed.

Keywords: Big data; data pipeline; data driven manufacturing

1. Introduction and background

Big data as defined by its characteristics is batched or streaming data that comes in high volume, high variety and high velocity [1]. To harness big data for a specific use, a variety of technologies (hardware and software) are integrated to enable data transmission, storage, processing and analysis [2].

Many situations in modern day manufacturing warrant the use of big data. For example, big data is needed for data-driven ad-hoc queries of the system [3], real-time process and quality monitory [4], machine learning [5] and designing digital twins [6].

The management of a big data project is best captured using its data pipeline, which describes the path through which data originates from source and goes through a variety of technologies to enable user applications [7]. Prior to implementing a data pipeline for a specific data driven project, a number of concerns arise. For instance, what should be the data source and data flow? Which technologies should be added to the pipeline? How should the technologies be seamlessly integrated to enable data flow? Which technology should be chosen? Meanwhile, a data pipeline design may be subjected to a number of iterative designs before a suitable one is realized. All these make the design process a challenging task.

Frameworks that have been advanced for data driven manufacturing are available [4, 8, 9] but they are not useful for guiding one on how to design a data pipeline. There are studies where researchers have presented big data architectures and big data flow maps for manufacturing systems. For example, [10] present a high-level architecture for the semantic analysis of complex events in manufacturing. [6] present a MTConect-based data flow from machine to user application. [11] propose a framework for quality prediction and operation control for metal casting, with emphasis on cyber physical systems. [12] used Integrated Definition diagram to describe their holistic approach to fault diagnosis in cyclic manufacturing processes. [13] promote a data science toolbox for industrial applications. Their toolbox maps the data flow through individual tasks in a machine learning analysis. These few examples are representative of the extant literature. We realized that architectures and data flow maps so far advanced in the literature, are not generalizable for the variety of data driven manufacturing projects. Additionally, it is unclear how manufacturing practitioners design their data pipelines since

10.1016/j.procir.2020.04.016

most researchers present a pre-designed data pipeline in the form of an architecture or framework, or simply report the steps that were followed in their data driven project.

While it is a challenge to design the big data pipeline for a specific data driven manufacturing project, there is a paucity of methods to navigate manufacturing practitioners through these challenges. Consequently, the aim of this research is to promote a framework that can be used for designing a broad range of data pipelines for manufacturing systems. The framework consists of a template for identifying the key layers and components in the data pipeline. Accordingly, the remainder of this paper is structured as follows: section 2 examines the literature to reveal key layers and components that make up a big data pipeline for manufacturing systems; section 3 is used to present the framework and explicate it using a manufacturing case while section 4 is the discussions and conclusions.

2. Key layers and components in a big data pipeline

Layers are those points along the data pipeline where data is exploited [9]. Components can be described as the individual technologies (such as hardware, software and algorithms) in each layer. Evidences from the literature suggest there are four fundamental layers that make up a big data pipeline [9, 14, 15]. We designate these layers as: a) data source; b) raw data transmission; c) data communication, processing and analysis tools and d) data visualization. We include a fifth, the source data types, as being an important element that determines the data pipeline layers and their components. These five layers cover the main issues that one would come across when undertaking a data driven project. Subsequently, we detail, with examples from the literature, each layer and the typical components for manufacturing systems.

2.1. Data sources

For clarity, we define a data source in terms of its root source. Under our definition, a database is not a data source but a data lake. Based on evidences from the literature, we found six main sources of big data in manufacturing systems. They include: sensors; cameras; measuring devices; Radio Frequency Identification (RFID); machine logs through Programmable Logic Controllers (PLCs) and operator logs.

Sensors are by far the most ubiquitous of all big data sources in a manufacturing system [4, 11, 13, 16]. This is not surprising due to the different types, functions and uses of sensors. Moreover, modern machines are equipped with miscellaneous sensors to enable automation and remote monitoring [17]. Cameras have been used to capture images of products along the production line, to monitor defects [18]. Measuring devices such as dynamometers have been used to acquire trust force in a drilling operation [19]. RFIDs (and barcodes) are printed labels that store information about an item (usually a product), to track its location and/or state in time and have been used by [20] in the analysis of job processing times.

Machine PLCs are designed to receive signals from input devices such as sensors, and use the inputs to control output devices such as switches. They can be considered as mini-computers that store data from input and output devices and so are able to generate data about machine parts and machine operating conditions [12, 21].

Situations may warrant machine operators and shopfloor employees to manually record data about machine and operational activities. [11] investigated a case where operators logged quality inspection data directly into legacy database systems. In a study by [22], shopfloor employees utilize barcode readers to collect data about process events through barcode tags placed next to machines.

2.2. Raw data transmission enablers

We define raw data transmission enablers as those technologies and devices that allow data to be moved from source to the point along the data pipeline where the data is first exploited. Examples in the literature include Internet of Things (IoT)-enabling microcontrollers [4, 16, 23]. We also noticed the prevalence of communication protocol software and hardware technologies which are used in harmonizing data from heterogenous machines, sensors and devices, typically found in factory settings. Examples in the literature include the use of Zigbee [16], MTConnect [6], OPC Unified Architecture [24], MC-SUITE [25] and Representational state transfer (REST) [6, 11]. In some situations, removable storage devices may be appropriate for use in transferring data from a PLC or microcontroller, to a computer that hosts the big data software tools [12].

2.3. Source data types

With evidences from the literature, data at source has been known to come in various forms mainly as lines of text [26]; eXtensible Markup Language [27]; JavaScript Object Notation [4, 25], frequency signals [28] and images [18]. These data types determine the technologies for data communication, processing and analysis. For example, image and frequency data often warrant the use of feature extraction algorithms, to extract only the useful attributes [18, 25].

2.4. Data communication, processing and analysis

There are different software tools for specific pipeline tasks [15] and their uses are varied. For data ingestion and communication, [4, 25] applied Kafka and Storm for data ingestion and publishing respectively, [6, 25] used REST application programming interface to pull data from source; [25] utilized custom codes to correct sparse sensor data; [6] used a MES to receive and store barcode data; [21] used HTTP protocol to standardize data from multiple sources; [26] applied Apache Pig and Hive to classify data and create relational tables for the data.

For storage, we found that most storage is cloud-based and in the form of non-relational, Non-Structured Language (NoSQL) format such as MongoDB [4], HBase [29] and CouchDB [25]. PostgreSQL, an object relational database which allows sensor data, image data and geographical maps to be stored in a data base, has been used by [16]. Legacy relational databases such as SQL [30] and Microsoft Excel [31] have also been used. The choice of data ingestion and storage is dependent on the familiarity with the technology and if the tool is appropriate for the project. Other evaluation criteria include: ability to handle batch and/or stream processing, scalability, fault tolerance, programming language used (i.e. the native interface such as python, Scala, java, R and scripting language to allow interactive shell support) and latency (the time it takes for data to be stored and/or retrieved, whether low or high-latency) [15].

Data exploration and analysis software tools are used to gain deeper understanding of the data and its characteristics. A number of software solutions have been applied for data analysis, exploration and modelling. [30] used Spark for feature extraction regarding different types of temperatures relating to a plastic injection molding operation, [32] used KNIME for data mining purposes in a predictive maintenance study, while [33] used KNIME to gain a better understanding of data. [34] applied Neo4J to label key features. [20] made use of TensorFlow to cluster RFID-based data about a shopfloor process. [26] used R programming software to create prediction models and classification algorithms to enable quality prediction. [18] used MATLAB for feature engineering.

In the literature, a variety of models and algorithms have been used, majorly for feature engineering and machine learning purposes. These include models for classification problems such as Decision Tree, Random Forest and Neural Network; models for clustering such as Support Vector Machine and Density-based spatial clustering of applications with noise (DBSCAN) [4, 12] and regression models for regression-based analysis [35].

Algorithms that have found widespread use in data driven manufacturing projects include Principal Component Analysis, k-Nearest Neighbour, Genetic Algorithm [18]. These algorithms are mainly applied in situations that warrant feature extraction, feature selection and dataset dimensionality reduction, for example when dealing with images and high frequency data which tend to have a high number of features.

Choosing which model or algorithm to use is dictated by the problem, but it is common to test different models and algorithms and then select the one with the most promising prediction performance result [11, 33]. The models and algorithms can also be used in a complementary fashion, for example [4] used DBSCAN to detect and remove outliers from the data, before using Random Forest for real-time predictive analysis.

The use of custom codes and web-based applications for enabling data visualization is common. [4] develop a web-based application for real-time data visualization utilizing JSON. [36] develop a web-based maintenance monitoring application utilizing the MIMOSA open system architecture. For remote monitoring of machine, [21] used a solution developed by the machine supplier. Cloud-based services were found to be the standard for developing and hosting data visualization software and programs, to enable remote data visualization

2.5. Computer hardware

Computers (desktop and laptops) and hand-held devices such as Tablets and smart phones [21] are widely used for data entry and visualization. Handheld devices can be used by shopfloor workers to monitor the process in real-time [36]. Desktop computers are mostly used as servers and hosting of big data software as well as cloud-based applications [4].

3. The proposed framework

Our framework for designing data pipelines for manufacturing systems is depicted in Fig. 1. It follows established convention of depicting frameworks for data pipelines, see [9, 14]. The framework shown in Fig. 1 delineates the key layers and components of data pipelines as described in the previous section. A guideline for using the framework is shown in Fig. 2. Subsequently, we present a use case to explicate how the framework can be utilized in the design of a data pipeline for a data driven manufacturing project.

The use case is an ink viscosity monitoring system for a gravure printing machine. For the type of operation, printing ink viscosity and print drying time are affected majorly by ambient temperature and humidity [37]. For uniformity in print quality and to prevent excessive ink and solvent usage, ink viscosity needs to be as consistent as possible during the printing operation. Prior to the study, machine operators needed to take manual viscosity readings every four to five minutes to monitor ink viscosity.

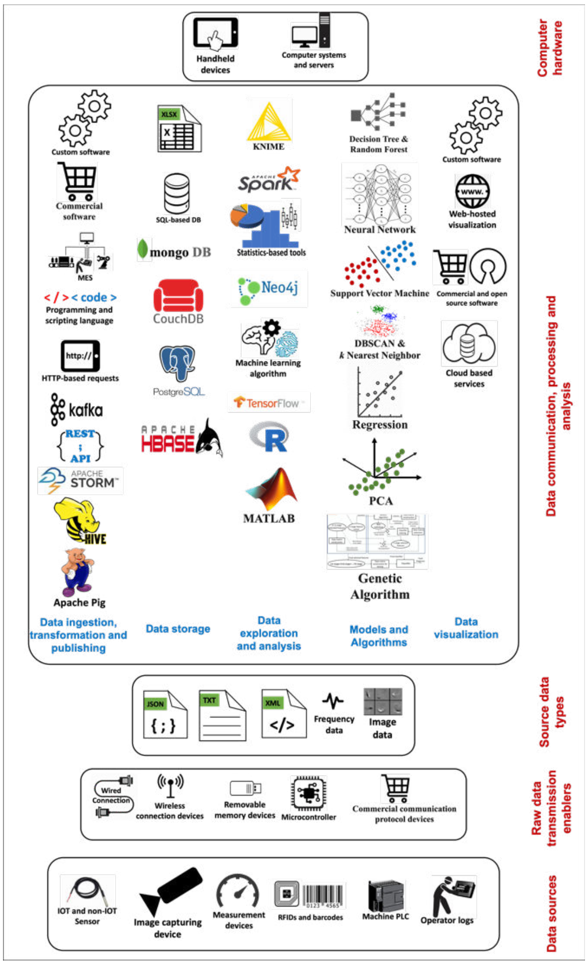

Fig. 1. Framework for designing big data pipelines in manufacturing systems

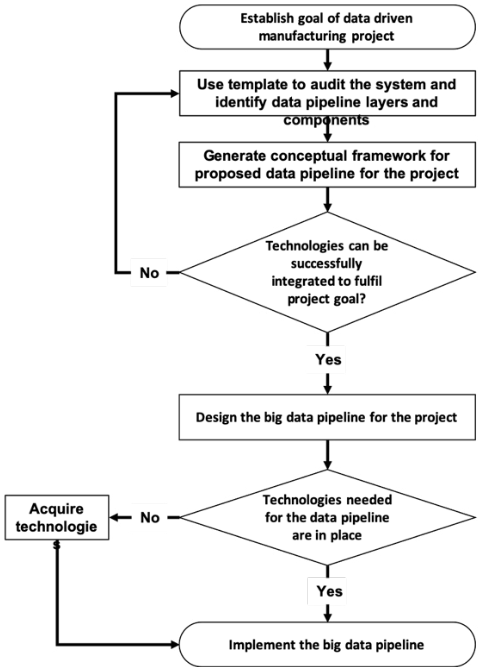

Fig. 2. Guideline for using the framework

3.1. Establish goal of data driven manufacturing project

It was necessary to first establish the goal of the data driven project, as this helps to scope the pipeline layers and components. The goal of the project as set out by the company was to generate a predictive model for monitoring ink viscosity by way of monitoring ambient temperature and humidity. This can be described as a regression or classification modelling problem, with two independent variables (temperature and humidity) and one dependent variable (ink viscosity).

3.2. Use template to audit the system

Using the template (Fig. 1) as a guage, we audited the system to determine what components were in place and those to be considered. There was no temperature and humidity measuring device. Based on the template options, we considered the use of an IoT-enabled sensor to allow wireless sensing. To facilitate this, we considered a microcontroller configuration for ease of installation and cost [4, 16].

The viscosity data was not being stored. We considered using a tablet-PC as being the most probable option to effectively collect operator-logged viscosity measurements. The tablet-PC can be mounted on the machine for close proximity to the operator. Microsoft Excel software was considered sufficient for creating and editing spreadsheets and files that would store the viscosity data. With the tablet-PC and spreadsheet, the machine operator could record the viscosity values in a structured data format with auto date/time stamp and auto save on every data entry. The date/time stamp of the operator log data was necessary to have a synchronized dataset when combined with the sensor data. In addition to the tablet-PC being used to collect viscosity data, it could also be made to communicate with the microcontroller to receive the sensor data. A wired or wireless communication would be possible to enable this.

The project on completion was to be transferred to the print manager, for replication in other systems in the factory. The print manager was not familiar with big data technologies and this had to be taken into consideration when selecting the data processing and analysis technologies. Using the template and references from the literature, we identified three candidate software: KNIME, Spark and MATLAB. KNIME, an open source big data software, was chosen for its graphical user interface and ease of use.

3.3. Generate conceptual framework

An audit of the system enabled us delineate the components (see Fig. 3) that could be considered for the project. Testing is necessary to check the success possibility of the integrated components. We were not able to test hardware since none of the considered hardware devices were in place in the system. We could however hypothesize the raw data type and test the software tools.

Open source experimental datasets exist for testing big data software technologies [38]. For the purpose of testing the data processing and analysis tools in our designed pipeline, we relied on a sample temperature and humidity dataset from another project we had undertaken: an excerpt of this dataset is show in Fig. 4a. This dataset came from a microcontroller-enabled sensor system, the type that was being considered for the project, see also [38]. For the viscosity measurements, we generated a hypothetical dataset. For this we used information provided by the operators since they were familiar with the readings. An excerpt of this dataset is shown in Fig. 4b.

Data cleaning is sometimes necessary because raw data rarely comes in a form that can be directly processed or analyzed by big data software tools. The raw dataset for temperature and humidity readings (Fig 4a) shows a semi-structured dataset, with no column headers or row numbers. This is an indication that the dataset would require cleaning, (for example, extracting only the important information such as time, temperature and humidity values) and manipulation (creating a structured dataset with column headers and serialized row numbers).

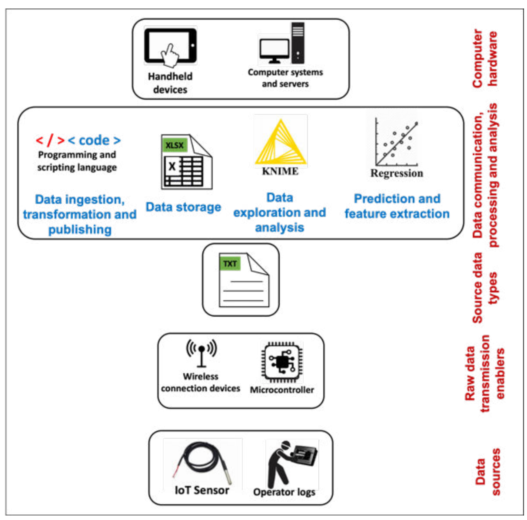

Fig. 3. Conceptual framework for proposed data pipeline

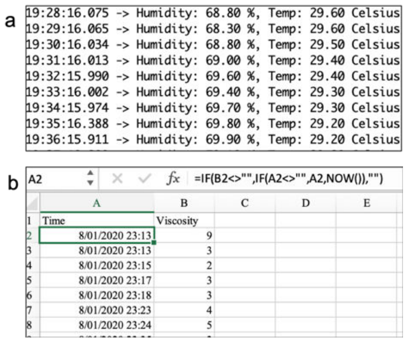

Fig. 4. (a) excerpt of sensor data; (b) excerpt of operator logged data

Data cleaning be achieved using a script written with programming language. As a result, we amended our initial data pipeline to include this component (the first data pipeline we generated had no component depicted for data ingestion, transformation and publishing).

3.4. Design the big data pipeline

On the basis of the audit (section 3.2) and the conceptual framework (section **Error! Reference source not found.**), we generated a final design of the data pipeline (Fig. 5), which shows the integrated components and data flow. To implement the data pipeline at a minimal cost, the following hardware components were acquired: for logging viscosity data, a Linx 12X64 - 12.5-inch Tablet-PC; Arduino Uno R3 microcontroller circuit board for programming the sensor to transmit data; Arduino compatible DHT22 temperature and humidity sensor; Arduino compatible DSD TECH HM-10 Bluetooth module to enable wireless communication between sensor and the tablet-PC; Power bank with 5v output supply pin to power the Arduino board. The software tools were KNIME and a python script.

3.5. Implement the big data pipeline

The acquired components were assembled according to the pipeline design. The IoT-enabled sensor configuration is shown in Fig. 6. This assemblage was placed close to the ink tray of the printing machine, to read the temperature/humidity around the ink tray area. The tablet-PC was mounted on the machine for the operator to input and record viscosity measurements. The microcontroller was connected wirelessly to the tablet-PC, using the Bluetooth module on the circuit. This allowed the microcontroller and tablet-PC to communicate, and by so doing enable the sensor to receive input signals through the tablet-PC, while allowing the tablet-PC to receive sensor data. The sensor was programmed to read temperature and humidity every five minutes, to allow as much synchronization with the rate of logging viscosity data and minimize the instances of missing data when the two datasets are combined based on time stamp attribute.

To test the assemblage, we used a small dataset generated from five consecutive production shifts. The sensor data was converted from TXT to XLSX format with labelled columns (date, time, temperature, humidity). A python script was run on the terminal window of the tablet-PC to enable this. The operator logged data was saved in XLSX format. The final dataset for the sensor data was a matrix of four columns and 704 rows. The dataset for the viscosity data was a matrix of three columns (date, time, viscosity) and 683 rows. Although these datasets are not considered big data by definition, the datasets were sufficient to test the data pipeline. Both datasets were transferred to a desktop computer that hosted the KNIME software. The download was by achieved by connecting the tablet-PC to the desktop though Universal Serial Bus cable.

KNIME software was used to build the prediction models. KNIME is able to process datasets in XLSX format, so there was no need for any further dataset transformation. Fig. 7 shows the workflow elements for a polynomial regression model. Similar workflows were generated for linear and logistic regression models, as well as Decision Tree predictor. The KNIME-based models functioned as expected, giving confidence that the data pipeline was a success and the project could be progressed to gather more data. The regression model with the most accurate prediction performance score would be selected and used to predict ink viscosity, using temperature and humidity readings around the ink tray.

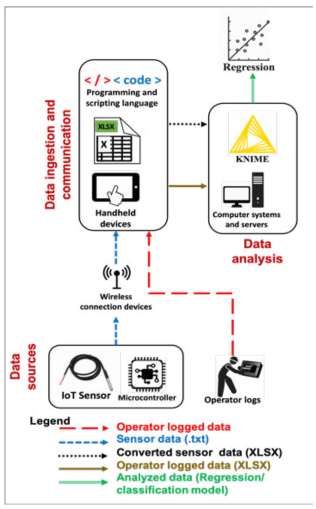

Fig 5. Data pipeline design

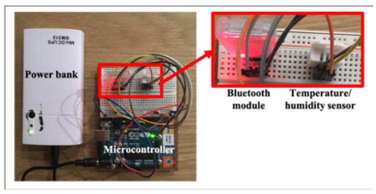

Fig. 6. IoT-enabled sensor

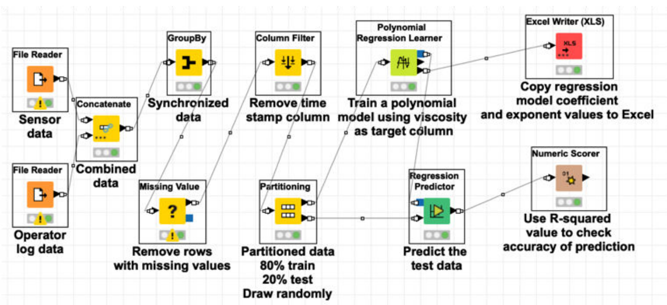

Figure 7. KNIME workflow for generating regression model

4. Discussions and conclusions

In this study, we develop and explicate a framework for designing big data pipelines for manufacturing systems. The framework was developed with generality in mind, and embodies the key steps (layers) that need to be traversed in a data driven manufacturing project. The framework template attempts to be exhaustive, and makes provision for the miscellaneous technologies (both hardware and software) that are needed to fulfill a data driven manufacturing project. Some technologies may have been missed, for example feature extraction algorithms which are of many different types. In the future, we plan to incorporate more technologies in the framework to improve its all-encompassing attribute.

We believe our framework has the potential to formalize the way data pipelines are designed and implemented, as described with the use case. We have plans to validate the framework for this purpose. The framework did consider qualitative aspects relating to a data driven project such as data science expertise level of project handlers, project cost and management commitment to the data driven project. These are aspects that we came across in the case study. In the future, these and other qualitative aspects would be considered as supplementary to the framework, thereby widening the scope of the framework. We also plan to expatiate on the layer components to provide users of the framework with in-depth information about the individual technologies. At a later date, we intend to utilize the framework to update the use case data pipeline for a feature engineering project for the same system. This would qualify our framework for use in updating an existing data pipeline. Data security continues to be a topical issue in data driven systems. In our future framework, we intend to include components relating to data security. Familiarity with big data technologies is advantageous when undertaking big data projects. It is the same with using our framework. We believe our framework (with the supplementary information contained in section 2) can offer inexperienced practitioners an entry level approach to their data driven projects.

References

[1] Oussous A, Benjelloun F, Lahcen A. Big Data technologies: A survey. J of King Saudi Uni Comput & Inf Sci 2018; 30:431-8.

[2] Geerdink BA. Reference architecture for big data solutions introducing a model to perform predictive analytics using big data technology. 8th IEEE-ICITST 2013:71-6

[3] Woo J, Shin S, Seo W. Developing a big data analytics platform for manufacturing systems: architecture, method, and implementation. Int J Adv Manuf Techn 2018;99:2193-217.

[4] Syafrudin M, Alfian G, Fitriyani N. Performance Analysis of IoT-Based Sensor, Big Data Processing, and Machine Learning Model for Real-Time Monitoring System in Automotive Manufacturing. Sensors 2018;18:2946-70.

[5] Ferreira, Silva R, Pereira V. Feature selection using non-binary decision trees applied to condition monitoring. 22nd IEEE-ETFA 2017:1-7

[6] Coronado P, Lynn R, Louhichi W. Part data integration in the Shop Floor Digital Twin: Mobile and cloud technologies to enable a manufacturing execution system. J Manuf Sys 2018;48:25-33.

[7] Pääkkönen P, Pakkala D. Reference architecture and classification of technologies, products and services for big data systems. Big data Res 2015;2:166-86.

[8] Zhang Y, Ren S, Lin Y. A framework for Big Data driven product lifecycle management. J Cleaner Prod 2017;159: 229-40.

[9] Tao F. Qi Q, Liu A. Data-driven smart manufacturing. J Manuf Sys 2018;48:157-69.

[10] Huber M, Voigt M, Ngomo A. Big data architecture for the semantic analysis of complex events in manufacturing. Informatik; 2016:353-60.

[11] Lee J, Noh S, Kim H. Implementation of cyber-physical production systems for quality prediction and operation control in metal casting. Sensors 2018;18:1428-45

[12] Kozjek D, Kralj D, Butala P. A data-driven holistic approach to fault prognostics in a cyclic manufacturing process. Proc CIRP 2017;63: 664-9.

[13] Flath C, Stein N. Towards a data science toolbox for industrial analytics applications. Comp Ind 2018; 94:16-25.

[14] Munshi A, Yasser A. Big data framework for analytics in smart grids. Elect Power Sys Res 2017;151:369-80.

[15] Ismail A, Truong H, Kastner W. Manufacturing process data analysis pipelines- a requirements analysis and survey. J Big Data 2019;6:1-26.

[16] Uhlmann E, Lagmouchi A, Geisert C. Decentralized data analytics for maintenance in industrie 4.0. Proc Manuf 2017;11: 1120-26.

[17] Yen I, Zhang S, Bastani F. A framework for IoT-based monitoring and diagnosis of manufacturing systems. IEEE-SOSE 2017:1-8.

[18] Chondronasios A, Popov I, Jordanov I. Feature selection for surface defect classification of extruded aluminum profiles. Int J Adv Manuf Techn 2016;83:33-41.

[19] Caggiano A, Angelone R, Napolitano F. Dimensionality Reduction of Sensorial Features by Principal Component Analysis for ANN Machine Learning in Tool Condition Monitoring of CFRP Drilling. Proc CIRP 2018;78:307-12.

[20] Kho D, Lee S, Zhong R. Big Data Analytics for Processing Time Analysis in an IoT-enabled manufacturing Shop Floor. Proc Manuf 2018;26:1411-20.

[21] Edrington B, Zhao B, Hansel A. Machine monitoring system based on MTConnect technology. Proc Cirp 2014;22: 92-7.

[22] Bevilacqua M, Ciarapica F, Diamantini C. Big data analytics methodologies applied at energy management in industrial sector: A case study. Int J RF Techn 2017;8:105-22.

[23] Li W, Kara S. Methodology for monitoring manufacturing environment by using wireless sensor networks and the internet of things. Proc CIRP 2017;61:323-8.

[24] Kirmse A, Kraus V, HoffmannM. An architecture for efficient integration and harmonization of heterogeneous, distributed data sources enabling big bata analytics. In ICEIS 2018;1:175-82.

[25] Ferry, N, Terrazas G, Kalweit P. Towards a big data platform for managing machine generated data in the cloud. In 15th IEEE-INDIN 2017:263-70

[26] Lee J, Yoon J, Kim B. A big data analytics platform for smart factories in small and medium-sized manufacturing enterprises: An empirical case study of a die casting factory. Int J Preci Eng Manuf 2017;18:1353-61.

[27] Lebold M, Richard K, Byington C. OSA-CBM architecture development with emphasis on XML implementations. In MARCON 2002:6-8.

[28] Amruthnath N, Gupta T. A research study on unsupervised machine learning algorithms for early fault detection in predictive maintenance. In 5th ICIEA 2018:355-61

[29] Xu W, Lin Q, Xu, W. Energy condition perception and Big Data analysis for industrial cloud robotics. Proc CIRP 2017;61:370-5.

[30] Lee B, Michaloski J, Proctor F. MtConnect-Based kaizen for machine tool processes. ASME Digital Collection 2010:1183-90.

[31] Persin S, Tovornik B, Muskinja N. OPC-driven Data Exchange between MATLAB and PLC-controlled System. Int J Eng Edu 2003;19:586-92.

[32] Massaro A, Maritati V, Gaialno A. ESB platform integrating KNIME data mining tool oriented on Industry 4.0 based on artificial neural network predictive maintenance. Int. J. Artif. Intell Appl 2018;9:1-17.

[33] Stanula P, Ziegenbein A, Metternich J. Machine learning algorithms in production: A guideline for efficient data source selection. Proc CIRP 2018;78:261-6.

[34] Nazarenko A, Sarraipa J, Camarinha-Matos L.Data Management for a Virtual Factory Collaborative Environment. Appl Sci 2019;9:4936-59.

[35] García V, Sanchez J, Rodrigues-Picon L. Using regression models for predicting the product quality in a tubing extrusion process. J Intell Manuf 2019;30:2535-44.

[36] Campos J, Jantunen E, Prakash O. A web and mobile device architecture for mobile e-maintenance. Int J Adv Manuf Techn 2009;45:71-80.

[37] EPA. The Effect of Ink Temperature On Solvent Losses and Print Quality. Available from: https///www.epa.gov/sites/production/files/2014-01/documents/gravure_effect_ink_temperature.pdf 2001:accessed 20 Dec. 2019

[38] https://www.kaggle.com/edotfs/dht11-temperature-and-humidity-sensor-1-day.