Distributed system for monitoring of welding processes and prediction of final products quality

Krzysztof Regulski[1], Łukasz Rauch[1], Krzysztof Bzowski[1],
Piotr Hajder[1], Monika Pernach[1], Jan Kusiak[1]
[1] AGH University of Science and Technology, Al. Mickiewicza 30, 30-059 Krakow, Poland
regulski@agh.edu.pl, lrauch@agh.edu.pl,
kbzowski@agh.edu.pl, phajder@agh.edu.pl,
pernach@agh.edu.pl, kusiak@agh.edu.pl

Keywords: high frequency induction welding, industry 4.0, process monitoring, non-destructive investigation

1. Introduction

Products welded linearly in high frequency induction process, like pipes or sheet-metal sections, are nowadays a major part of the market of customers from construction, furniture and mining industry, where the most important products are bolts and bended elements. These products are usually crucial from safety point of view, therefore online monitoring and investigation of their quality is of the highest importance. Currently testing of products mechanical properties is performed only selectively. This approach is mutual for all the companies producing welded section, but it does not guarantee that all products are tested and free of defects. The main problem of online investigation is significant velocity of the manufacturing process and its conditions precluding stable measurements. Up till now, there is no company in the world which implemented fully automated non-destructive investigation of products quality and control of the process. Thus, the main objective of the project was development of the hybrid system allowing non-destructive online investigation based on sensors monitoring manufacturing devices supported by numerical procedures, analyzing stability of the process and predicting properties of the welded products. The set of monitoring sensors were equipped with thermovision and optical cameras, Hall-effect measuring devices and others. Numerical procedures will be realized by using data mining and machine learning methods mapping state of the process and product properties. Realization of such process maps required series of laboratory experiments, which were performed in one of the project stages. Additionally, sensitivity analysis was applied after experiments and maps creation to confirm, which parameter of the process has the highest influence on the properties of the product. The proposed approach is convergent with augmented intelligence [1–3], supporting human work as a domain expert with experience of highly qualified engineers.

2. System architecture

2.1. Main idea of the system

The proposed system is based on Industry 4.0 idea integrating monitoring of sensors reading process parameters and modelling of the products quality on the basis of numerical models or metamodels. The software design includes a physical data acquisition module – one at each aggregate

The publication is co-financed from the state budget under the programme
of the Minister of Education and Science called “Excellent Science” project no. DNK/SP/548041/2022

Doskonała Nauka
Ministry of Education and Science Republic of Poland
Republic of Poland

(main production device). The data are exchanged between the aggregate and a server with storage module. The main task of the data acquisition module is to operate the software that communicates directly with the PLCs and other sensors (monitoring devices) operating at an aggregate. Each type of sensor has a separate communication interface, for which software had to be implemented. The system design includes the ability to communicate with:

- PLCs using UNIX sockets (binary telegrams),
- an industrial printer using UNIX sockets (XML messages),
- MSSQL database - start-stop (ODBC communication),
- defectomat (limited, local ODBC communication),
- Basler high-speed camera (communication via dedicated device SDK),
- scanCONTROL line scanner (communication via dedicated device SDK).

Each communication module receives a data packet from the device, performs data analysis and/or executes local transformation if required, and then publishes it to the Apache Kafka communication bus. Apache Kafka is a message broker capable of handling real-time data from multiple nodes, offers stream processing mechanisms and allows multiple technologies to be combined into a cohesive and reliable information system. According to the nomenclature of the Kafka environment, each communication module is a data producer. Data is read from the broker by consumers (programmes listening to messages published on the broker). In the system, the main data consumer is the TimescaleDB database, a PostgreSQL-based solution designed to handle time series.

TimescaleDB provides number of functionalities focused on continuous aggregation and filtering of data using materialised views (MATERIALISED VIEW). A materialised view is a type of special table that stores the results of a query aggregating data. The primary use for materialised views is in the results of spatial analyses, which require a lot of time and computational power to compute. Materialised view data is retrieved from persistent storage (disk), and is not generated on demand like classic SQL views.

2.2. Visualization and main Graphical User Interface

The user interfaces is divided into main part dedicated for managers and module for operators. The main view is refreshed at 1-minute intervals and includes data changed in the last hour. If the view is used to aggregate actual data, aggregation will be performed on demand. Similar views have been implemented for each device from which the system acquires data. In order to speed up readings, the data in the database has been split into historical and current data. Current data is stored on a fast SSD and historical data is moved to a slower HDD. Data older than one month is considered historical. The data archiving task has been implemented as a Postgres database stored procedure and is run daily.

Data presentation is handled by a dedicated web service implemented using the NodeJS environment and the NestJS library in TypeScript. The communication architecture with the application's browser client is based on GraphQL technology. GraphQL is a flexible way of communicating between the client and server application, in which the client decides what data it needs to perform a given task in the user interface. Communication is based on a central access point and is distinguished by a strongly typed schema (communication is based on data in a well-defined type). GraphQL supports a subscription mechanism based on WebSockets, enabling real-time communication. This functionality has been used to develop a view of both the current operating status of the selected aggregate and the items currently produced. The client application subscribing to the events is informed in real time of any change on the server side. This treatment was dictated by the lack of stable support for implementing message consumption mechanisms directly from the Apache Kafka broker in web browsers. The client application was implemented using the React library and adapted to work in web browsers. The design of the client application was divided into 4 panels. An example of the application's screen content is shown in Figure 1.

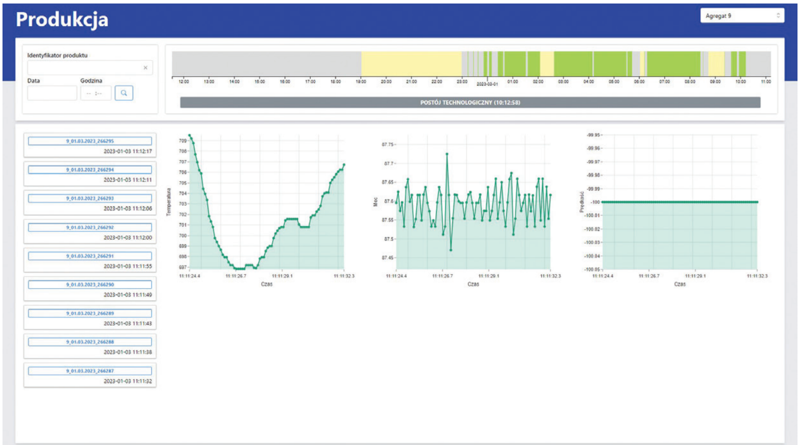

Figure 1. The main interface for users managing and monitoring production.

2.3. Support of operators

Part of the system supporting operators of the process is built on WebWorker and WebAPI architecture – in the background, data is downloaded using the hardware manufacturer's library functions, at a set sampling rate, processed, filtered and sent to a message broker (Apache Kafka) from where it is transferred to the database. The application client has been implemented using the React library with unattended use in mind. This means that it will attempt to retransmit data intermittently every 60 seconds if sensor reading errors are detected or a failure occurs or there is a power failure. This decision was dictated by the requirement for implementation on the production floor, where the prevailing conditions are not conducive to operating the application using peripherals such as a mouse or keyboard. The user interface was implemented with modularity and flexibility in mind. Data from each sensor is displayed in a separate independent component, making system development and maintenance easy. Also, failures of individual devices do not bring the client application to a halt. The connection of each visualization component to the corresponding data source is asynchronous and independent of the others. If data cannot be displayed, the component will display an appropriate error message and attempt to establish a connection until successful. An example of working operator's GUI is shown in Figure 2.

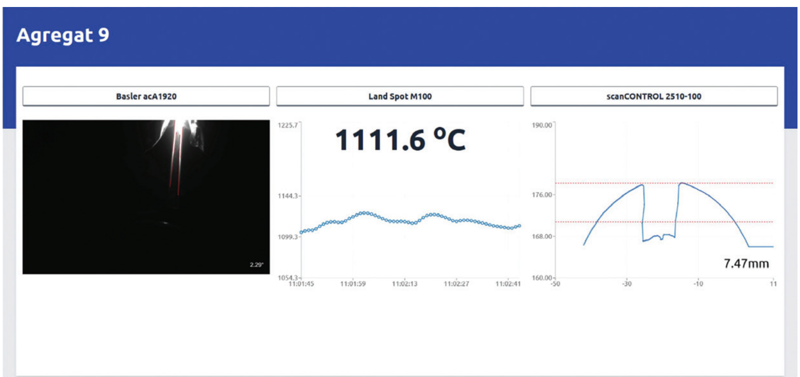

Figure 2. Additional graphical interface for users operating the process.

References

1. Jan Z., Ahamed F., Mayer W., Patel N., Grossmann G., Stumptner M., Kuusk A.: *Artificial intelligence for industry 4.0: Systematic review of applications, challenges, and opportunities*. Expert Systems with Applications, 216, 2023, 119456.
2. Alicioglu G., Sun B.: *A survey of visual analytics for Explainable Artificial Intelligence methods*. Computer&Graphics, 102, 2022, 502–520.
3. Arrieta A.B., Díaz-Rodríguez N., Del Ser J., Bennetot A., Tabik S., Barbado A., Garcia S., Gil-Lopez S., Molina D., Benjamins R., Chatila R., Herrera F.: *Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI*. Information Fusion, 58, 2020, 82–115.

Acknowledgements. The work is financed by National Centre for Research and Development within a project no. POIR.01.02.00-00-0091/19.