FIREMAN

WP3 Deliverable 3.2
Design and Functional Architecture for Data Acquisition

Project Title:	Framework for the Identification of Rare Events via MAchine learning and IoT Networks

Title of Deliverable:	Design and Functional Architecture for Data Acquisition
Status-Version:	Final-1.0
Delivery Date:	31/7/2020

Contributors:	Daniel Gutiérrez Rojas, Pedro Nardelli, Charalampos Kalalas, Ioannis Christou, Constantinos Papadias
Reviewers:	Francisco Helder Candido (UFC), Elvis Stancanelli (UFC), Richard Demo Souza (UFSC)
Approved by:	All Partners

Document Revision History

Version	Date	Description
v0.1	30 April 2020	Table of Contents
v0.2	14 June 2020	Main contents added
v0.3	23 June 2020	Version delivered for internal review
v0.4	05 July 2020	Version delivered for external review
v0.5	24 July 2020	Addressed comments from external review
v1.0	31 July 2020	Approved form of Deliverable D3.2

Summary

This document summarizes the progress made so far in the context of local data collection and storage procedures in WP3. The consortium partners have introduced several approaches for local data compression at sensor level which are compatible with the system architecture design proposed in WP2. Aiming to address the fundamental challenges of capturing highly heterogeneous and big-volume data from the physical processes, the research outcomes in Deliverable 3.2 are expected to provide a first pre-processing framework towards large-scale data acquisition in industrial environments. The research is performed within the context of the EU CHIST-ERA project FIREMAN.

Table of Contents

1 Introduction

1.1 Objective of the document

The objective of Deliverable 3.2 is to compile and document in a coherent manner the research outputs that have been carried out in **Task 3.2: Local data collection and storage** within the context of project FIREMAN. Task 3.2 aims to provide an initial pre-processing and storage framework towards large-scale data acquisition in industrial setups to be further developed. This framework is expected to address highly-heterogeneous industrial datasets characterized by a large volume of information. In this context, the consortium partners have devised local data compression techniques for reduced use of communication (focus of Deliverable 3.1) and storage resources while aiming to address the fundamental challenges for local data acquisition at sensor level.

1.2 Structure of the document

The remainder of the document is organized as follows. Section 2 provides a functional description of the system architecture adopted in FIREMAN and elaborated in WP2. Section 3 summarizes the main challenges encountered during local data acquisition in Industrial Internet-of-Things (IIoT) environments. Section 4 presents the different approaches proposed by the consortium partners for local data compression at sensor level. Section 5 presents an analysis of the possible choices for database selection for the storage needs of the project, which is empirically supported by the experiences of other related projects concurrently running with FIREMAN. Finally, Section 6 provides our concluding remarks.

2 System Architecture Design

The system architecture proposed in WP2 is designed to be general and highly flexible, therefore it may be applied to different scenarios and cases. The key idea is to assess Cyber-Physical Systems (CPS) using three interrelated layers, namely physical, data, and decision. This approach was initially proposed in [1, 2] to assess the dynamics of physical systems that are regulated based on a decision-making process that depends on data processing, but was simply focused on theoretical toy-models. Figure 1 depicts the FIREMAN proposed extension, together with the underlying communication network topology. The proposed general framework is based on key questions that must be answered to determine the particular rare-event solution to be designed, which is the key goal of the project. The idea is to define the viable options that we need to consider to have an effective solution for particular industrial processes, as well as practical limitations imposed by the industry itself (e.g., preference for private networks, or already deployed wired communication systems).

The questions are structured in steps that follow the three-layer model mentioned above. The first step is to identify the rare event(s) under consideration, also considering whether the problem is known beforehand. To have a quantitative evaluation of the related physical processes related to the event, sensors are needed to map the physical to the data layer. Here one key question is what kind of sensors should be used? How many should be used and where should they be located? After the locations are confirmed, the next phase is the sampling strategy from those sensors: it may be periodic, event-driven (non-periodic), or a mix between them. We then need to determine the time granularity and/or the event that triggered a sampling. The next phase is to define the communication system to be used. In particular, the type of access technology (wireless or wired), the network (internet or private network) and storage (local database, cloud, private cloud). Once the data is stored, data should be aggregated, as other variables are stored with the same timestamp. Depending on the information of the monitored variables, some of them could be suppressed. The question here is how data should be clustered/aggregated/structured/suppressed (fused)? The information gathered after the data fusion will be used to detect the event when adding variables that monitor the physical condition of the grid. Here the question that arises is how to make the rare event detection ultra-reliable in our problem, keeping in mind that data needs to be suppressed with extreme care.

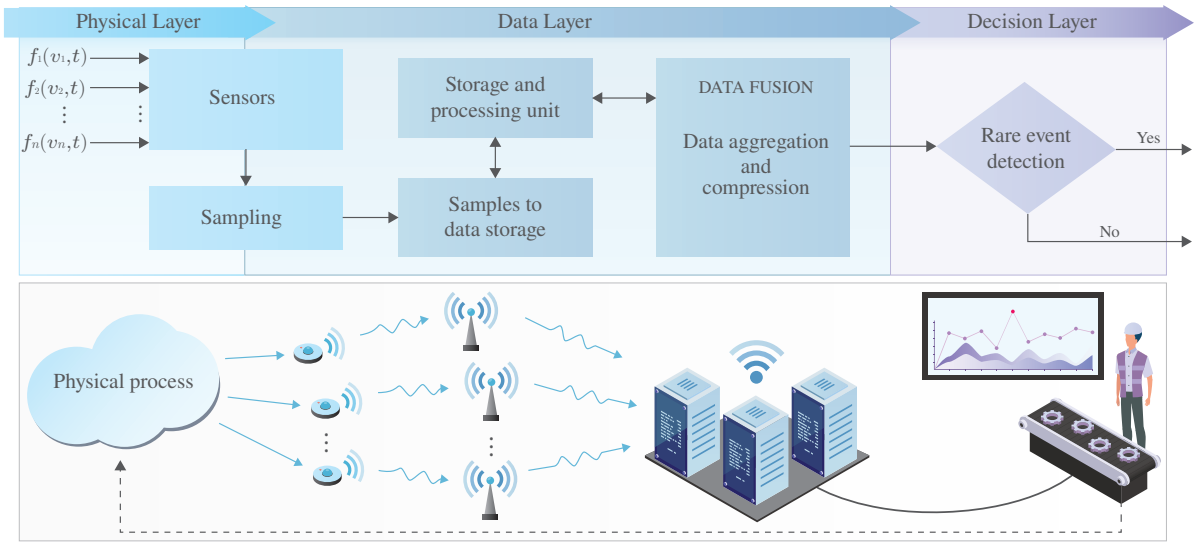

Figure 1: Three layer framework for rare event detection [3].

3 Challenges on Local Data Acquisition

The three-layer framework indicates the importance of data pre-processing already in the acquisition phase in order to avoid issues related to the traffic offered to the communication networks (Deliverable 3.1) and to storage. However, the limited storage, processing and computational capabilities of sensors installed in IIoT equipment, introduce fundamental challenges for local data acquisition and pre-processing which need to also account for the residual energy levels in the IIoT sensors' operation. The direct application of conventional large-scale data acquisition approaches that can be computationally complex to implement is not therefore straightforward [4]. In addition, due to dense deployment of IIoT sensors and the high amount of energy required to maintain constant communication with the data aggregator, every local data acquisition strategy needs to consider a compressed representation of the captured set of measurements [5]. In this context, data compression helps deal with the challenges of data storage, collection, transmission, processing, and analysis at a local level [6]. On the other hand, data compression is often lossy, i.e., the reduction in the data volume often comes at the cost of degraded reconstruction accuracy. Besides instructing sensor sampling and transmission schemes, properly designed local data compression techniques would allow the optimal configuration of the underlying communication protocols which will be part of Task 3.3 research activities. In particular, injecting only relevant data in the network would help reduce the connectivity overhead and prevent the under-utilization of the scarce radio resources. Local data acquisition strategies are also expected to impact the cluster formation methods developed in Task 4.1 since a lower volume of transmitted data would directly impact the number of required relays to extend the limited communication range of IoT sensors.

Given the aforementioned challenges, our goal is to exploit the spatial/temporal structure of the sensor measurements to devise lightweight compression techniques with minimum reconstruction error. In particular, we aim at revealing interrelations between the seemingly independent heterogeneous sensor streams and come up with local data compression tech-

niques that leverage the potential pairwise correlations while keeping the detection of abnormal state conditions unaltered. Our computationally efficient schemes focus at reducing the communication overhead and the energy consumption in resource-constrained IIoT environments empowered by wireless connectivity.

4 Local Data Compression Strategies

This section presents two approaches for data compression. One is related to *temporal compression* for each time-series. This method is dynamic and real-time. The compression capabilities of this method are studied in the context of the Tennessee Eastman Process (TEP), presented in the Appendix. The other method is based on spatiotemporal correlation between measurements of different sensors, utilizing established signal processing techniques. The compression using this method is based on IEC-61850 traffic data from substation automation related to an indicative power grid use case.

4.1 Data Compression for Event-driven Sampling

The main idea of an event-driven approach for this application is to perform data compression in order to transmit the meaning of information from the data acquisition point to the data fusion point. This approach was described in Deliverables 2.3 and 3.1. It consists of the following steps: (i) input data from all 52 sensors (N); (ii) variable average estimation and margin selection (90% of lowest/highest values) from normal operation; (iii) at every time slot (k) for each variable, if the values are out of the margins the sample is transmitted, otherwise, if nothing is received at the data fusion point, the variable will maintain the average value estimated from the previous step; (iv) compression rate calculation for each variable. The limit values for margin selection for each variable were chosen arbitrarily. An example of this approach is seen in Figure 2 where the signal obtained from the sensors is shown with its limits. The samples transmitted are only the ones that are out of the upper and lower limits as seen in Figure 3. This setup allows to transmit less data via any communication system. In the example mentioned above, the compression rate is 92.60%, which means that only about 7.4% of the samples are transmitted. The pre-processed time series based on the proposed event-driven method will serve as input to the data fusion and analytics, where anomalies should be detected, identified, and diagnosed.

Figure 4 shows the compression ratio for all the variables in the baseline case with different percentages of thresholds. The compression ratio increases as the threshold gets closer to the max value of each variable. It can be seen from Figure 4 that the compression ratio is a non-linear function of the threshold that is selected, therefore the selection of the most suitable threshold is not straightforward, and will depend on the end-application to be defined.

We also evaluated another event-driven method based on "delta" (the variation from one sample to the other) introduced in Deliverable 3.1. The operation principle can be described as follows: (i) the maximum variation from sample to sample from a normal operation state is calculated; (ii) we then check in the fault files whether the variation at data acquisition point is less than a percentage of that max variation and, in case this happens, (iii) the sample value is not transmitted and we maintain the previous value that was sent at the fusion point. The same variable depicted previously, but using this so-called "delta" approach can be seen in Figure 5 and its compression ratios in Figure 6.

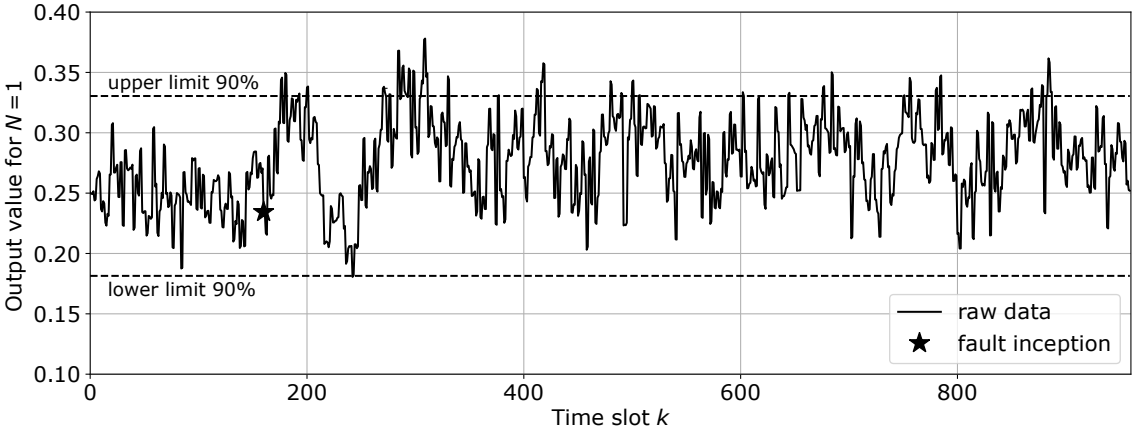

Figure 2: Signal from variable 1 at data acquisition point (before transmission) for fault number 2 of Tennessee Eastman dataset.

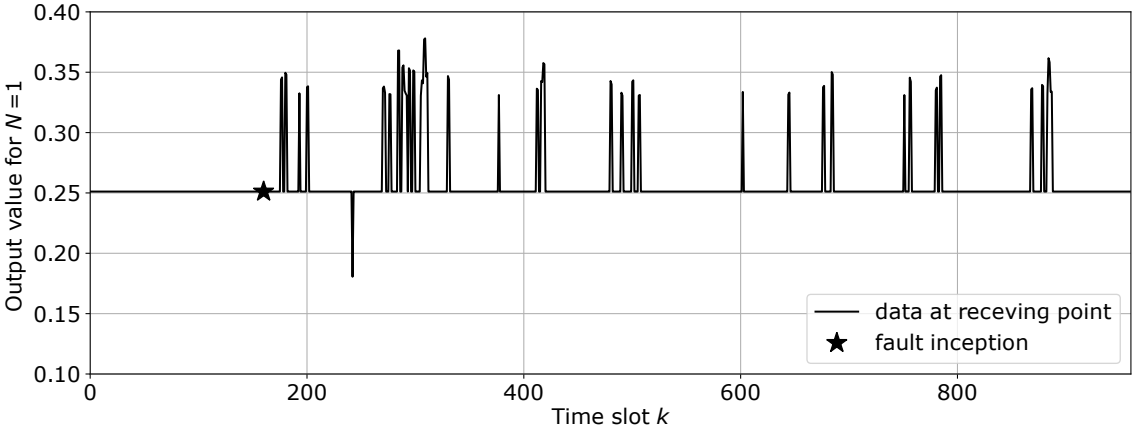

Figure 3: Signal from variable 1 at data fusion point (after transmission) for fault number 2 of Tennessee Eastman dataset, using the threshold approach.

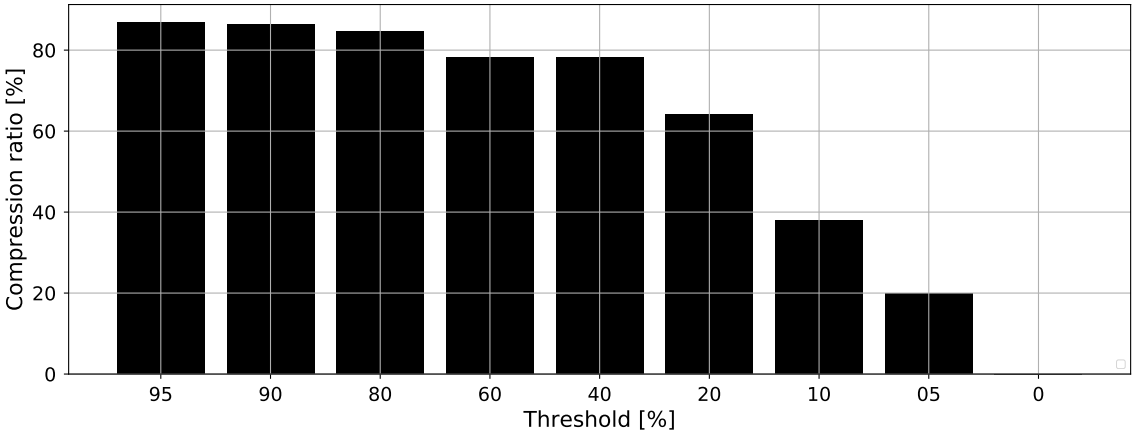

Figure 4: Compression ratio at different threshold percentages.

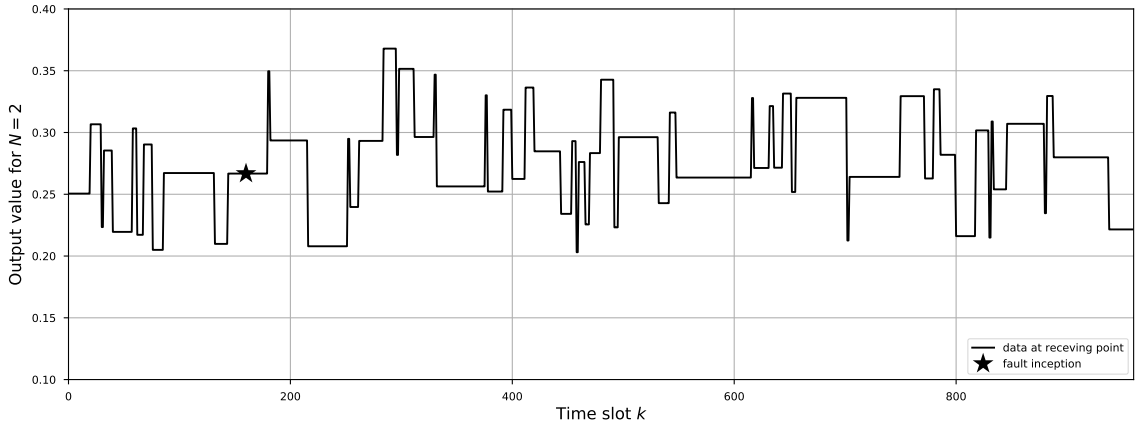

Figure 5: Signal from variable 1 at data fusion point (after transmission) for fault number 2 of Tennessee Eastman dataset, using the delta approach.

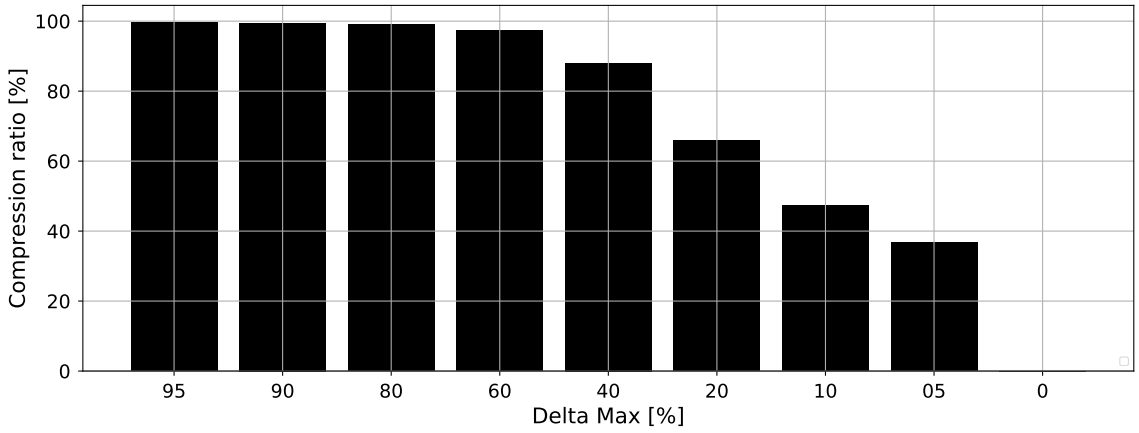

Figure 6: Compression ratio at different delta percentages.

4.2 Data Compression for Spatiotemporal Sensor data

In [7], we have proposed a data reconstruction scheme able to *i*) learn the hidden patterns of time-series sensor data which exhibit high spatiotemporal correlation; and *ii*) mine their dynamics, in order to adequately handle missing sensor observations due to the imperfect wireless transmission. We use a probabilistic model for linear dynamic systems [8], as follows:

$$\mathbf{z}_{t+1} = A\mathbf{z}_t + \mathbf{w}_t, \tag{1}$$

$$\mathbf{y}_t = C\mathbf{z}_t + \mathbf{v}_t. \tag{2}$$

where $\mathbf{y}_t$ denotes the sensor measurement at time t and $\mathbf{z}_t$ is the corresponding hidden variable at time t. In Eq.(1) and Eq.(2), A denotes the linear hidden state transition matrix (temporal continuity) and C the linear projection matrix from the latent variables to the sensor observations. Using an iterative coordinate descent procedure, we have derived an estimate of the expectation of the missing measurements conditioned on the observed ones, using a sequence of latent variables to model the dynamics and hidden patterns of the observation sequence [7].

Besides recovering the missing observations with minimum reconstruction error, our proposed scheme can be also suitable for time-series compression tasks. Therefore, we have

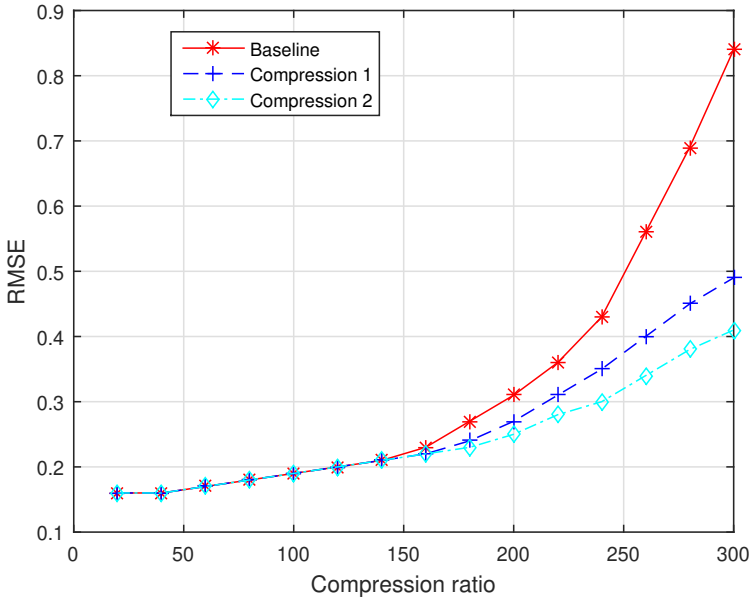

Figure 7: Performance comparison in terms of RMSE for three different compression techniques using the IEC-61850 dataset.

evaluated its applicability to local data compression at sensor level. Given the limited storage capabilities at each sensor, instead of storing the direct sensor observations, we store part of the latent variables learned during the time steps of the proposed scheme. In addition, we store the matrices A and C as well as the covariance matrices Q and R of the $\mathbf{w}_t$ and $\mathbf{v}_t$, respectively. By properly tuning the hidden dimension (which constitutes a model hyper-parameter) as well as the number of time ticks of the stored latent variables, we can easily identify a tradeoff between compression ratio and error. In what follows, we have considered two different compression approaches and evaluated their performance in terms of decompression error. In particular:

1. The first compression method will first learn the hidden variables and will store them for every k time ticks.

2. The second compression method will first learn the hidden variables and will store them for only the *necessary* time ticks, i.e., the time steps where the computed error is greater than a given threshold.

The rationale behind the second method is to efficiently store the sensor measurement data without compromising the detection performance for abnormal behaviours that may exist in the sensor streams. We compare the proposed methods with a baseline approach that combines Singular Value Decomposition (SVD) and liner interpolation as follows: *i*) it first projects the data into principle dimensions using SVD; *ii*) it stores the hidden variables for k time ticks and the projection matrix from SVD; *iii*) it projects back the hidden variables using the stored matrix while the gaps are filled using linear interpolation.

In Figure 7, we demonstrate the (de-)compression performance in terms of root mean square error (RMSE) for the aforementioned approaches. As in [7], the considered dataset

consists of IEC-61850 network traffic traces generated based on open-source libraries for protocol implementation[1] and a discrete-event network simulator [9]. The dataset exhibits high levels of spatial and temporal correlation across the sensor measurements to model the real behaviour of substation automation systems, e.g., local cascade failures result in inter-dependent transmissions of measurements from neighboring devices. For all examined methods, the compression ratio is defined as the ratio of the total uncompressed space over the compressed space. It can be observed that our proposed compression methods outperform the baseline one, especially for high compression ratios. In addition, the second approach, which selectively stores the hidden variables based on the resulting error, is able to achieve even better compression performance.

[1] Available online at: https://github.com/mz-automation/libiec61850

5 Database Selection for Big Data Handling

In addition to compression strategies, it is important to define the database to be used to store the data. Storing sensor data on a Relational Database Management System (RDBMS) is an option that is quickly becoming as appealing as the option of storing data directly on flat files. While most RDBMSs (e.g., Oracle or MySQL or PostgreSQL) offer full Atomicity, Consistency, Isolation and Durability (ACID) properties, they are not capable of inserting documents at the speeds of document (json) stores or key-value stores.

The advantages of using high-throughput high-availability document stores such as MongoDB have been recently demonstrated [10]. These advantages, at a high level, include:

1. Consistency;
2. Partition tolerance;
3. Ease of data replication in multiple servers;
4. Easy sharding of data among multiple servers.

At a more technical level, storing large CSV files as single text string to be later manipulated by libraries such as pandas –in Python– is trivial computationally, and still very efficient in terms of the required underlying database operation [10]. For a particular use case data, reading the entire dataset from MongoDB using pymongo has been shown to be between 50% and 100% faster than reading them from a MySQL server through the standard MySQL JDBC connector. The particular dataset comprised of about 40 thousand rows and around 40 columns, fully dense.

Similarly, since the data (sensor readings) are meant to be almost never updated once written in the database storage areas, and since complex queries on these data are not required by machine learning or data mining algorithms, using a store whose data manager pays more attention to throughput than concurrency control has a strong advantage over a classical RDBMS model that also incurs the overheads of SQL plan construction etc. Regarding API and API usage, MongoDB offers excellent bindings to all popular programming languages for the project including C/C++, Java, Node.js, Python, Julia and R. In addition, MongoDB has excellent connectors to two of the most popular distributed computing infrastructures that are likely to be of great use within the project and in later industrial settings as well: Apache Kafka and Apache SPARK.

5.1 Using Apache Kafka as Data Store?

Due to the popularity of Apache Kafka as a universal message bus, offering publish/subscribe functionalities that are highly scalable, several projects and systems have opted for Apache Kafka as the main data store, advocating the use of "data streaming" in an ever-changing world of infinite data. We do not subscribe to this approach, and prefer instead to publish our own philosophy on the matter, which is that data stores are not message buses, and message buses are not data stores. Data stores have specific advantages that Kafka does not have and will never possess (in the same way that Kafka has certain advantages that data stores do not have). Consistency of the stored data, throughput, and advanced query capabilities including

map-reduce capabilities inside the database server, are among the most important features of MongoDB that Kafka cannot –and should not– offer.

Combined together with the superior performance that MongoDB offers both while reading sensor measurements as well as while writing them for the first time on a collection, choosing MongoDB becomes the best choice for the project, with a second choice being Oracle 18c (which is also free for development purposes but requires commercial license for actual production use).

6 Conclusions

The first step towards large-scale data acquisition in IIoT environments resides at the sensor level. The deployed sensors are typically characterized by limited storage, processing, and computing capabilities; therefore, data compression appears as a natural choice for reduced data handling. In this context, Deliverable 3.2 aims to consolidate the research activities in the context of Task 3.2 related to "**Local data collection and storage**". A description of the functional system architecture was provided along with an elaboration of the local data compression approaches adopted by the consortium partners, aiming for reduced use of communication and storage resources. This Deliverable is expected to provide valuable insights for the work carried out in other WPs and, in particular, serve as an input for subsequent data compression tasks performed at a higher level of the system architecture.

Appendix

Tennessee Eastman model

The dataset material consists of several faulty cases of an industrial plant, as produced by the Tennessee Eastman problem [11]. The process has five major equipments, namely a condenser, a vapor-liquid separator, a reactor, a product stripper, and a recycle compressor (Fig. 8). Its objective is to obtain the products G and H from the reactants A, C, D and E. This is reached by a set of four chemical reactions, in which components B and F are, respectively, an inert and a byproduct. More details can be found in [12, 13]. This benchmark is suitable to evaluate process monitoring schemes and control strategies based on data driven analysis. Besides the normal operation, 21 abrupt or incipient faulty conditions caused by common disturbances in practice are simulated [14]. There are 52 monitoring variables or features, being 11 manipulated variables and 41 measured variables. Once a faulty condition occurs, all are generally affected with changes in their respective values.

The Tennessee dataset was generated in a process simulator that has been widely used by the process monitoring community[2]. It is composed by 22 subsets named *dXX_te.dat*, where $XX = 0, 1, 2, \cdots, 21$. The file *d00_te.dat* refers to the normal operating condition. Each of the other ones relates to a particular fault, that is, a different shift from this reference condition. The subsets consist of 960 observations of the 52 variables, which are sampled every 3min with a Gaussian noise. The faults are introduced after 8 simulation hours.

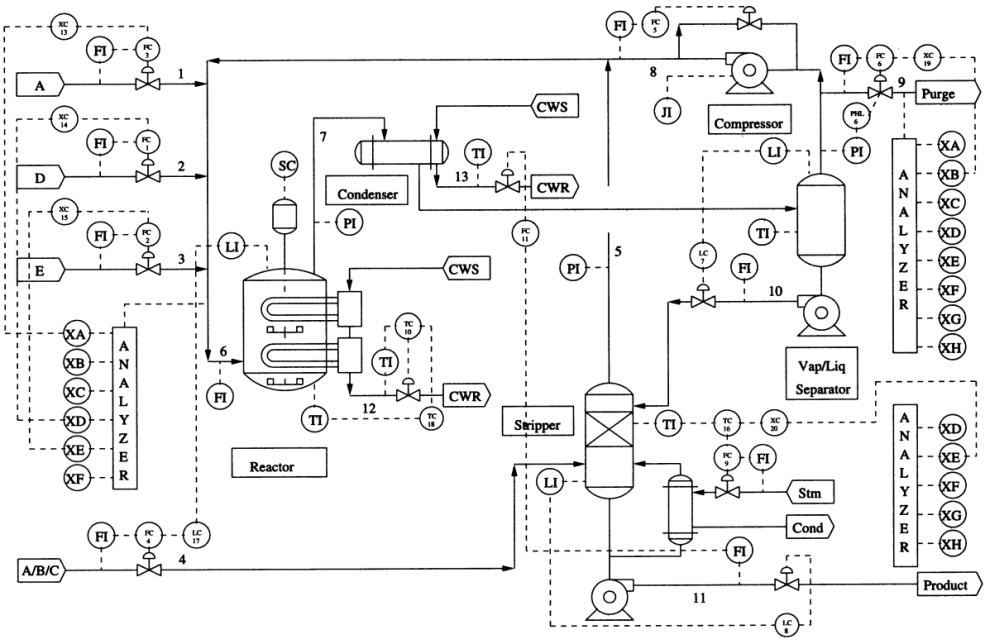

Figure 8: Process flow diagram of the Tennessee Eastman problem [13].

[2] https://github.com/camaramm/tennessee-eastman-profBraatz

References

[1] F. Kühnlenz and P. H. Nardelli, "Dynamics of complex systems built as coupled physical, communication and decision layers," *PloS one*, vol. 11, no. 1, p. e0145135, 2016.

[2] P. H. Nardelli and F. Kuhnlenz, "Why smart appliances may result in a stupid grid: Examining the layers of the sociotechnical systems," *IEEE Systems, Man, and Cybernetics Magazine*, vol. 4, no. 4, pp. 21–27, 2018.

[3] D. Gutierrez-Rojas, M. Ullah, I. T. Christou, G. Almeida, P. H. J. Nardelli, D. Carrillo, J. M. Sant'Ana, H. Alves, M. Dzaferagic, A. Chiumento, and C. Kalalas, "Three-layer Approach to Detect Anomalies in Industrial Environments based on Machine Learning," 2020.

[4] D. B. Rawat, J. J. P. C. Rodrigues, and I. Stojmenovic, *Cyber-Physical Systems: From Theory to Practice*. USA: CRC Press, Inc., 2015.

[5] N. Kimura and S. Latifi, "A survey on data compression in wireless sensor networks," in *International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume II*, vol. 2, 2005, pp. 8–13 Vol. 2.

[6] S. Li, L. D. Xu, and X. Wang, "Compressed Sensing Signal and Data Acquisition in Wireless Sensor Networks and Internet of Things," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 4, pp. 2177–2186, 2013.

[7] C. Kalalas and J. Alonso-Zarate, "Sensor data reconstruction in industrial environments with cellular connectivity," in *2020 IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (IEEE PIMRC '20)*, August 2020.

[8] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009.

[9] C. Kalalas, J. Alonso-Zarate, and G. Bag, "On the Transmission Mode Selection for Substation Automation Traffic in Cellular Networks," in *IEEE International Conference on Smart Grid Communications (SmartGridComm)*, October 2017, pp. 1–7.

[10] EU PROPHESY Project, "https://prophesy.eu."

[11] J. J. Downs and E. F. Vogel, "A plan-wide industrial process control problem," *Computers and Chemical Engineering*, vol. 17, no. 3, pp. 245–255, 1993.

[12] S. Yin, S. X. Ding, A. Haghani, H. Hao, and P. Zhang, "A comparison study of basic data-driven fault diagnosis and process monitoring methods on the benchmark tennessee eastman process," *Journal of Process Control*, vol. 22, no. 9, pp. 1567 – 1581, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0959152412001503

[13] L. H. Chiang, E. L. Russell, and R. D. Braatz, *Fault detection and diagnosis in industrial systems*. Springer, 2001.

[14] E. L. Russell, L. H. Chiang, and R. D. Braatz, "Fault detection in industrial processes using canonical variate analysis and dynamic principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 51, no. 1, pp. 81–93, 2000.