


Article

Prediction-Correction Techniques to Support Sensor Interoperability in Industry 4.0 Systems

Borja Bordel ¹, Ramón Alcarria ^{2,*} and Tomás Robles ¹

¹ Information Systems Department, Information Systems School, Campus Sur, Universidad Politécnica de Madrid, 28031 Madrid, Spain; borja.bordel@upm.es (B.B.); tomas.robles@upm.es (T.R.)

² Department of Geospatial Engineering, School of Surveying Engineering, Campus Sur, Universidad Politécnica de Madrid, 28031 Madrid, Spain

* Correspondence: ramon.alcarria@upm.es

Abstract: Industry 4.0 is envisioned to transform the entire economical ecosystem by the inclusion of new paradigms, such as cyber-physical systems or artificial intelligence, into the production systems and solutions. One of the main benefits of this revolution is the increase in the production systems' efficiency, thanks to real-time algorithms and automatic decision-making mechanisms. However, at the software level, these innovative algorithms are very sensitive to the quality of received data. Common malfunctions in sensor nodes, such as delays, numerical errors, corrupted data or inactivity periods, may cause a critical problem if an inadequate decision is made based on those data. Many systems remove this risk by seamlessly integrating the sensor nodes and the high-level components, but this situation substantially reduces the impact of the Industry 4.0 paradigm and increases its deployment cost. Therefore, new solutions that guarantee the interoperability of all sensors with the software elements in Industry 4.0 solutions are needed. In this paper, we propose a solution based on numerical algorithms following a predictor-corrector architecture. Using a combination of techniques, such as Lagrange polynomial and Hermite interpolation, data series may be adapted to the requirements of Industry 4.0 software algorithms. Series may be expanded, contracted or completed using predicted samples, which are later updated and corrected using the real information (if received). Results show the proposed solution works in real time, increases the quality of data series in a relevant way and reduces the error probability in Industry 4.0 systems.

Keywords: Industry 4.0; interpolation techniques; predictor-corrector algorithms; data series; sensor nodes



Citation: Bordel, B.; Alcarria, R.; Robles, T. Prediction-Correction Techniques to Support Sensor Interoperability in Industry 4.0 Systems. *Sensors* **2021**, *21*, 7301. <https://doi.org/10.3390/s21217301>

Academic Editor: Juan M. Corchado

Received: 29 July 2021

Accepted: 29 October 2021

Published: 2 November 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The strengthening of important global crises, such as the climatic crisis or the natural resource crisis, makes essential a change in the productive schemes of all countries, but especially in those with a relevant industrial sector [1]. The increase of efficiency in all industrial production processes is the only solution to optimize the use of resources, support the citizens' wellbeing and strengthen social development [2]. Industry 4.0 is an innovative paradigm referring to this new era [3].

In Industry 4.0, production systems and solutions implement mechanisms to make flexible, automatic and real-time decisions [4] that guarantee the adaptation of production processes to the variable behavior of economic, social and physical contexts [5]. With this approach, the global efficiency of industry has proved to increase significantly [6]. Paradigms, such as cyber-physical systems [7] or artificial intelligence [8], are basic to enable this new era, although all these monitoring mechanisms and decision-making algorithms are supported by a common technology: sensor nodes [9].

Using the sensor data, high-level software modules may create models to represent (and later predict) the production processes' and industry context's evolution, make real-time business decisions and (even) trigger alarms about workers' safety and wellbeing [10].

Some of these activities are critical in industry, and a precise evaluation and stable, high-quality, physical monitoring are essential to avoid fatal problems. At the software level, algorithms commonly match these requirements. However, sensor nodes are much less stable, and they present many random malfunctions [11].

Sensing platforms may be affected by human actions (including hits, blurs, misalignments, etc.), numerical errors (especially in digital sensors measuring derivate variables, such as the electrical power), hardware problems (such as overheating, aging, inactive periods, etc.), embedded software issues (for example, memory congestion, blocking instructions, etc.) and communication malfunctions (such as variables delays, jitter, packet losses, interferences, etc.), among many other potential impacts [12]. Although all these malfunctions are not relevant if long-term analyses are performed (e.g., for statistical applications), they may cause a critical situation if an inadequate decision is made in real time based on a low-quality data flow. Currently, the way in which this problem is commonly addressed is by considering a seamless integration of sensor nodes and high-level software components [13], integrating calibration phases and algorithms for adjusting or compensating effects, such as delays, jitter or numerical errors, etc.

Nevertheless, this seamless integration highly affects the social and economic impact of the Industry 4.0 paradigm, as only some (or even only one) sensor technologies may be employed in each application, as calibration models, compensation algorithms, etc. [14], are totally dependent on the specific sensors and processing algorithms to be integrated. This turns Industry 4.0 into a very rigid and close paradigm, more similar to a proprietary solution than to a flexible, open approach. Thus, the adoption of the Industry 4.0 paradigm may get very slow because of its high cost (caused by a lack of competitiveness in the market) and its low interoperability.

To address this challenging situation, new solutions guaranteeing the interoperability of all sensor nodes and sensor technologies with every possible software element and processing algorithm in Industry 4.0 solutions are needed.

Therefore, in this paper, we propose a general solution to ensure the high quality of data flows and their adaptation to the algorithms' requirements, valid for all sensor technologies and algorithms. It is a flexible and adaptable solution that may be integrated in every Industry 4.0 system. The proposed solution is based on numerical algorithms following a predictor-corrector architecture. Given a data flow to be curated and adapted, first, using a combination of techniques, such as Lagrange polynomial and Hermite interpolation, a set of potential, curated data series is calculated. Later, the most probable series is selected according to the statistical properties of the historical series. The original data series may be expanded, contracted or completed using predicted samples, which are later updated and corrected (second phase) using the real information (if received).

Contrary to other proposals (based, for example, on Gaussian distributions), this scheme is not dependent on the sensor technology or the software modules to be integrated, and it introduces (as shown in Section 4) a negligible delay, so real-time operation is not affected (something essential in Industry 4.0 systems). As a result, the sensor platform may interoperate with any high-level application, with no adaptation or calibration procedure. The produced curated time series has enough quality to be integrated with any kind of software module.

The structure of the paper is as follows: Section 2 presents the state-of-the-art on sensor interoperability in Industry 4.0 scenarios. Section 3 describes the main proposal, including the mathematical foundations. Section 4 includes an experimental validation analyzing the performance of the proposed solution. Finally, Section 5 shows the conclusions and future work.

2. State of the Art on Sensor Interoperability in Industry 4.0 Scenarios

In the last 10 years, the idea of "sensor interoperability" has been understood in many ways [15]. From some approaches focused on hardware compatibility [16], to other views related to the cyber-physical system revolution and focused on abstract and high-level

issues, such as synchronization [17]. However, the most modern and accepted definition for sensor interoperability was proposed by the Institute of Electrical and Electronic Engineers (IEEE) [18]: “the ability of two or more systems or components to exchange information and to use the information that has been exchanged”.

In this context, two basic aspects are identified within the challenge of sensor interoperability: the interconnection technologies and the adaptation technologies [19]. The first ones are focused on enabling the exchange of information, while the second ones aim to allow the use of the exchanged data.

One of the basic contributions to interconnection technologies is interoperability standards and architectures. The Industrial Internet Reference Architecture (IIRA) [20] was introduced in 2015 and provides a design process to integrate interoperability mechanisms in industrial Internet systems. On the contrary, the reference architecture model for Industry 4.0 [21] (also proposed in 2015) includes a set of guidelines to understand previously existing generic interoperability standards in the context of Industry 4.0. Nevertheless, the main topic investigated within this topic is cloud and edge architectures [22]. In cloud architectures, the interoperability problem is decomposed into elemental problems that are solved and executed in a distributed manner [23]. Although these architectures are very flexible and can be employed in all kinds of Industry 4.0 scenarios [24], they implement a star topology where all transactions must go through the cloud, which may cause bottlenecks and congestion under some circumstances [25]. Contrary to these traditional cloud architectures, the proposed solution may be adapted to different Industry 4.0 architectures. It can be deployed in edge architectures, just distributing all the independent modules among the different devices. But the proposed framework also allows several orchestrated instances working in one unique scenario, as the statistical model does not need a global understanding of the physical platform. Then, a mesh architecture can also be supported, where bottlenecks and congestion are easier to manage.

Other and heterogenous interconnection technologies have been also reported. Some proposals define models to fill the gap between low-level infrastructures and data analytics components [26], while other schemes integrate semantic web components and ontologies to connect cyber-physical systems and knowledge management modules [27]. Among all these solutions, digital twinning is the most promising approach [28]. Digital twins are comprehensive digital representations of physical components [29], so they can simulate the behavior of real underlying platforms through realistic models [30]. Using these twins, different enhanced interconnection middleware (for example, based on publication/subscription networks) have been tested in several application scenarios [31]. Although these mechanisms successfully interconnect all layers in an Industry 4.0 system, they cannot protect the high-level applications from errors or corrupted data in the infrastructure. Adaptation technologies are then required.

Regarding adaptation schemes, although data management and processing are one of the most popular, challenging and interesting topics currently [32], most reported Industry 4.0 solutions in this area are focused on high-level applications: from fault diagnosis [33], fault prediction [34] or prognosis [35] to semantic mechanisms [36] or data-fusion approaches [34,37]. However, in all these proposals, sensing data are considered to show the required quality, and no data curation or quality improvement mechanisms are described.

One of the key challenges addressed in the context of adaptation technologies is sensor reliability [38]. In autonomous systems, such as cyber-physical systems, problems, such as lost-data packages and data collision, must be addressed [39]. In general, however, proposed solutions are not focused on data curation but on analyzing how reliable the received data are. Different models to estimate the sensor reliability at any time have been reported: from the traditional, specific models for each sensor (typically motion sensors) [40] and probabilistic graphs [41], to modern machine-learning frameworks [42] or response filters [43] that operate with generic devices. The final objective of all these models is to enrich the decision process based on the collected data. Although promising

results have been reported, this approach needs the high-level, decision-making modules to be adapted as well, so the Industry 4.0 implantation costs and barriers tend to be higher. With the proposed solution in this paper, this challenge is addressed.

In semantic architectures, data adaptation is also critical. Different mechanisms to adapt and transform the different semantic standards into any other data format may be found [44]. Besides, ontologies to allow semantic data processing have been reported [45]. Contrary to the proposed solution, these schemes cannot be employed to protect the Industry 4.0 system against corrupted data, malfunctions, etc.

On the other hand, data-curation mechanisms are not explicitly addressed, as a seamless integration among hardware and software components [34,46] is the preferred approach in the literature. Hard and complex calibration processes are usually considered [6] to make the processing algorithms aware of the sensor nodes' biases. Besides, computationally heavy schemes to compensate different effects (such as redundant data) based on previous observations and offline processing may be found [47]. However, all these proposals do not enable sensor interoperability (they are totally application-specific); on the contrary, they make it difficult. Moreover, they are not flexible or dynamic solutions and, of course, they cannot be executed in real time (essential requirements in Industry 4.0).

Only a few proposals on actual data curation have been reported. In this area, most contributions are focused on outlier detection [11]. Using different techniques, datasets are transformed, and anomalous data are removed. Techniques based on digital encoders [48], machine learning [49], statistical indicators [50], performance indicators [16] or hybrid approaches [51] have been described. Although these schemes are useful, they cannot be employed in real time, and many other potential malfunctions, such as packet losses, cannot be addressed through these solutions. On the other hand, mechanisms based on signal-processing techniques may be found [15]. In these solutions, data are understood as communication signals, and they are curated based on instruments, such as the complex envelope. This approach may operate in real time and may correct and curate all kinds of malfunctions; however, it only considers one criterion to propose a curated data series. Thus, the error introduced by the curation algorithm is very variable, depending on how similar the sensor data under curation to a communication signal is. In some Industry 4.0 scenarios, this error may be too high to be acceptable.

Finally, some generic proposals on data analysis may be employed to support data curation in Industry 4.0. For example, algorithms to classify time series in an automatic and more flexible manner [52] have been reported. If only two labels (valid and invalid) are defined, this scheme could be employed for data curation. However, it cannot be employed in real time, and it does not enable the correction of errors in data series.

Contrary to all these previous proposals, the solution described in this paper may operate in real time, as it only operates with a limited amount of data. It is flexible and adaptable to all scenarios, as it does not depend on the sensor technology or software algorithms to be employed. Besides, all kinds of malfunctions can be curated, and up to four different potential curated data series are analyzed before selecting the most probable one.

3. Proposed Predictor-Corrector Solution

In this section, the proposed data duration mechanism, based on a predictor-corrector approach, is presented. Section 3.1 describes the general statistical framework to calculate and obtain the curated data series. Section 3.2 presents the different approaches to calculate the actual curated data flows, even in real time. Section 3.3 describes the models to analyze and estimate the different data malfunctions that may appear in Industry 4.0 solutions. Finally, Section 3.4 presents the mechanisms to update the curated data series if real data from Industry 4.0 is received in the future.

3.1. General Mathematical Framework and Curation Strategy

Given an Industry 4.0 platform \mathfrak{T} (1), a set of N different generic sensor nodes \mathcal{S}_i are generating N independent data series $y_i[n]$. These data series suffer different malfunctions,

and they are received by high-level software modules as a different set of N data series $x_i[n]$. These malfunctions are represented as a collection of L different functions λ_l (2) transforming the original series generated by the sensor nodes $y_i[n]$ into the received time series $x_i[n]$.

$$\mathfrak{T} = \{\mathcal{S}_i, i = 1, \dots, N\} \quad (1)$$

$$x_i[n] = \lambda_1 \circ \dots \circ \lambda_l \circ \dots \circ \lambda_L(y_i[n]) \quad (2)$$

Although other sensing patterns could be applied in industrial scenarios, samples are periodically generated and sent to the high-level software modules for real-time monitoring and decision making. Samples in the \mathcal{S}_i sensor node are produced each T_i seconds (3).

$$x_i[n] = x_i(nT_i) \quad n \in \mathbb{N} \quad (3)$$

Statically, each data series $x_i[n]$ is a realization of a stochastic process ϕ_i (4), where Ω is the universe of possible values ω_k generated by the sensor node \mathcal{S}_i . This universe is a subset of the field of real numbers \mathbb{R} (5). This universe is discrete and strictly depends on the hardware capabilities of the sensor node, and it is analyzed and reported by manufacturers.

$$\phi_i(n, \omega_k) = x_i[n] \quad \omega_k \in \Omega \quad (4)$$

$$\Omega = \{\omega_k \mid k = 1, \dots, K\} \subset \mathbb{R} \quad (5)$$

For each different time instant n_0 , the stochastic process ϕ_i transforms into a different random variable $X_i[\omega]$ (6) with some specific statistical properties.

$$\phi_i(n_0, \omega) = X_i[\omega] \quad (6)$$

However, in Industry 4.0 scenarios, physical variables evolve much slower than the sampling period T_i ; i.e., the superior frequency of physical signals f_{max} is much lower than the sampling frequency f_s^i (7).

$$f_{max} \ll \frac{1}{T_i} = f_s^i \quad (7)$$

In this context, for any time instant n_0 , it is possible to define an open time interval B_s around n_0 with radius ε (8), where the random variables show equivalent statistical properties for all time instants. Thus, we are assuming the stochastic process ϕ_i is locally first-order stationary in B_s (9).

$$B_s = \{n \in \mathbb{N} : d(n, n_0) < \varepsilon\} \quad (8)$$

being $d(n, n_0) = \sqrt{(n - n_0)^2}$ the Euclidean distance in \mathbb{R}

$$\phi_i(n, \omega) = \phi_i(n + n_c, \omega) \quad \forall n, n + n_c \in B_s \quad (9)$$

Given a data series $x_i[n]$, if data in the time interval $[n_1, n_2]$ should be curated, an expanded time interval $[n_1^e, n_2^e]$ (10) must be considered, so it contains the original interval $[n_1, n_2]$ where data must be curated, but it is included in the open time interval B_s (the stochastic process must be stationary in the interval).

$$[n_1, n_2] \subset [n_1^e, n_2^e] \subset B_s \quad (10)$$

This time interval $[n_1, n_2]$ may refer to a past time period (11) (so we are performing an offline data curation), but we can also perform a real-time data curation if the current time instant n_0 belongs to the time interval $[n_1, n_2]$ under study (12). If operating in real time, the proposed curation solution is employed as a prediction mechanism for calculating future samples in advance. If offline data curation is performed, the algorithm may be run as fast as possible, while in real-time data curation, the process is synchronized with

the sampling period T_i , so one sample is curated at each time instant, although as many samples as desired may be predicted with each new sampling period T_i .

$$n_0 \notin [n_1, n_2] : n_0 < n_1 \quad (11)$$

$$n_0 \in [n_1, n_2] : n_1 \leq n_0 < n_2 \quad (12)$$

The problem of data curation is to find a new time series $x_i^*[n]$ in the interval $[n_1, n_2]$, so it represents in a more precise way (compared to the original time series $x_i[n]$) the real situation of the Industry 4.0 system, represented by time series $y_i[n]$. As many random effects impact this study, a probabilistic approach is the most adequate, so this condition transforms in a comparison between two different probabilities, p_i^* and p_i (13). Hereinafter, $P(\cdot)$ is the probability function, calculating the probability of a predicate to be true.

$$p_i^* > p_i$$

$$p_i^* = P(x_i^*[n] = y_i[n] \quad \forall n \in [n_1, n_2]) = P(x_i^*[n]) \quad (13)$$

$$p_i = P(x_i[n] = y_i[n] \quad \forall n \in [n_1, n_2]) = P(x_i[n])$$

Figure 1 shows the proposed algorithm to find that time series $x_i^*[n]$, fulfilling the previous conditions (13), if it exists.

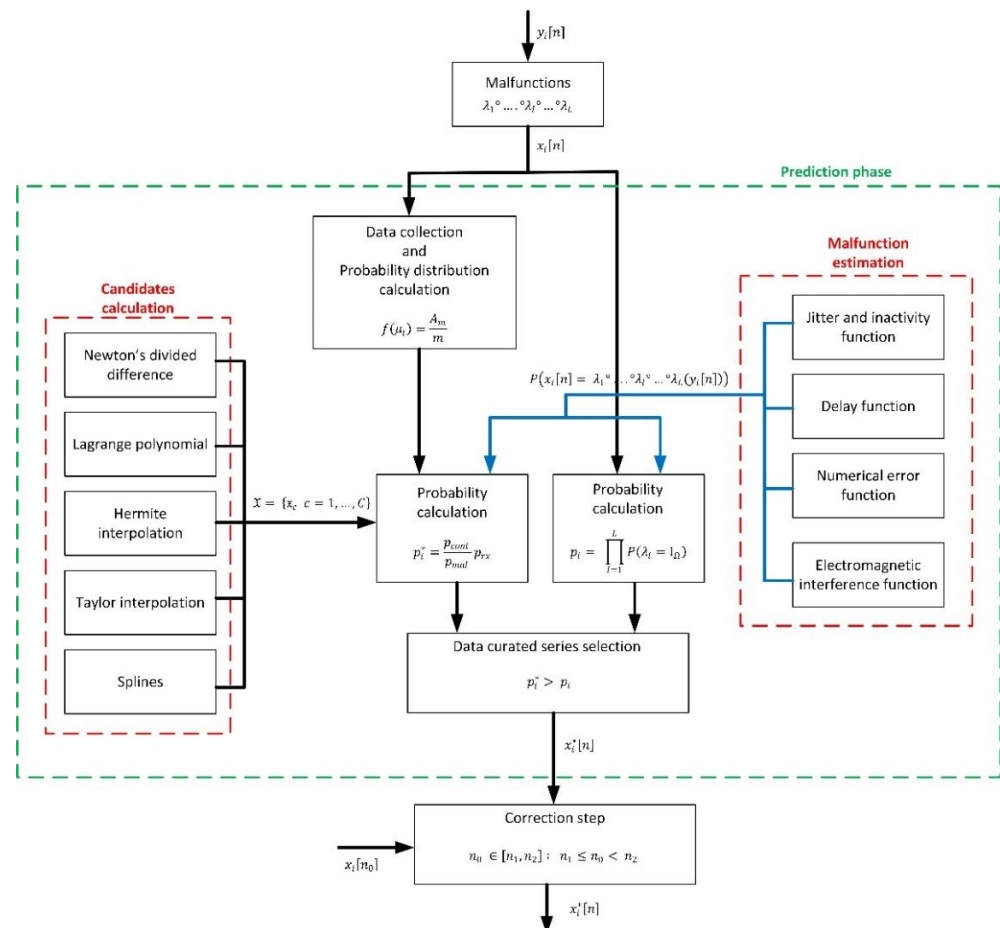


Figure 1. Global overview of the proposed prediction-correction algorithm.

As can be seen (in the initial prediction phase), first, a set of C suitable candidates \mathfrak{X} to be that curated time series $x_i^*[n]$ are calculated (14).

$$\mathfrak{X} = \{x_c, c = 1, \dots, C\} \quad (14)$$

To calculate those candidates, different techniques are employed, based on interpolation mechanisms. In particular, five different techniques are considered: Newton's divided differences, Hermite interpolation, splines, Taylor interpolation and Lagrange polynomial. The purpose of this approach is to guarantee the curated time series $x_i^*[n]$ is continuous and coherent with samples outside the curation interval $[n_1, n_2]$. Using the previously curated data in the expanded interval $[n_1^e, n_2^e]$, a collection of possible curated time series $x_i^*[n]$ are calculated, considering all samples define a continuous function. For each one of these candidates \mathfrak{X}_c , then, it is applied the statistical theory (Bayes' theorem) to obtain probability p_i^* . If, for any candidate, the curation condition (13) is met, that candidate \mathfrak{X}_c is selected as the curated time series $x_i^*[n]$. On the contrary, and depending on how different probabilities p_i^* and p_i are, the time series may remain as is, or the curate data series may be obtained as a combination of the most probable candidates and the original data flow $x_i[n]$.

If a real-time data curation is performed, new information about the curation interval $[n_1, n_2]$ is received at each sampling period T_i . In that case, a correction phase is carried out. In this phase, the new sample is compared to the predicted one, and (depending on how different they are) different actions are taken to correct the curated time series initially calculated.

To solve this problem, both probabilities p_i^* and p_i must be obtained.

Probability p_i represents the fact that the received data $x_i[n]$ are exactly those data generated by the sensor nodes $y_i[n]$. In other words, no malfunction (of any kind) has occurred in the interval $[n_1, n_2]$. In our model, that means functions λ_l are the identity function all of them (15). As all the malfunctions are physically independent, they are also statistically independent, and the joint probability may be rewritten as a product of unidimensional probabilities (16). Section 3.3 analyzes how to evaluate those probabilities for each one of the considered malfunctions.

$$p_i = P(x_i[n] = \lambda_1 \circ \dots \circ \lambda_l \circ \dots \circ \lambda_L(y_i[n]) = y_i[n]) = P(\lambda_l = I_\Omega \quad \forall l = 1, \dots, L) \quad (15)$$

$$p_i = \prod_{l=1}^L P(x_i[n] = \lambda_l(y_i[n]) = y_i[n]) = \prod_{l=1}^L P(\lambda_l = I_\Omega) \quad (16)$$

On the other hand, probability p_i^* is more complicated to calculate, and the Bayes' theorem is employed (17).

$$p_i^* = \frac{P(x_i^*[n] \mid x_i[n] \quad \forall n \in [n_1^e, n_2^e])}{P(x_i[n] \quad \forall n \in [n_1^e, n_2^e] \mid x_i^*[n])} P(x_i[n] \quad \forall n \in [n_1^e, n_2^e]) = \frac{p_{cont}}{p_{mal}} p_{rx} \quad (17)$$

To apply this theorem, three different probabilities, p_{cont} , p_{mal} and p_{rx} , must be obtained. Probability p_{mal} is the probability of functions λ_l (representing the malfunctions) to transform data series $x_i^*[n]$ into series $x_i[n]$ in the interval $[n_1, n_2]$. As said before, this probability may be written as a product of L different unidimensional probabilities (18). Section 3.3 analyzes how to evaluate those probabilities for each one of the considered malfunctions.

$$p_{mal} = P(\lambda_l(x_i^*[n]) = x_i[n] \quad \forall l = 1, \dots, L \quad n \in [n_1, n_2]) = \prod_{l=1}^L P(\lambda_l(x_i^*[n]) = x_i[n]) \quad (18)$$

Probability p_{rx} is the probability of receiving the sequence $x_i[n]$ in the interval $[n_1^e, n_2^e]$. That variable may be easily calculated using the probability function of the random variable (and stochastic process) $\phi_i(n, \omega)$ in the time interval B_s (19). In this case, once again, we are considering samples are independent events, so the join probability may be rewritten as a product.

$$p_{rx} = \phi_i(n, x_i[n] \quad \forall n \in [n_1^e, n_2^e]) = \prod_{n \in [n_1^e, n_2^e]} \phi_i(n, x_i[n]) \quad (19)$$

Finally, probability p_{cont} is the probability of the Industry 4.0 system's evolution to follow a continuous and coherent flow. In this case, we are evaluating how probable is the data series $x_i^*[n]$ to show certain values in the interval $[n_1, n_2]$, considering the other data received in the expanded time interval $[n_1^e, n_2^e]$. As the stochastic process is stationary in the interval B_s , the probability distribution g_i^0 the interval $[n_1, n_2]$ and distribution g_i^e in the expanded time interval $[n_1^e, n_2^e]$ must be identical. As both distributions become different, the probability of series $x_i^*[n]$ to be the best candidate for the curated series is reduced.

To calculate how different these two distributions are, we are employing the traditional function scalar product and the Lebesgue integral (20). However, in this case, as the universe under study is discrete, the Lebesgue integral may be approximated by a common sum. Thus, and considering the distance function induced by the function scalar product, we can calculate the distance d_g between distributions g_i^0 and g_i^e (21). Finally, to calculate the probability p_{cont} , we must apply a function transforming values in the interval $[0, \infty)$ in the interval $[0, 1]$ (22).

$$\langle g_i^0, g_i^e \rangle = \int_{\Omega} g_i^0 \cdot g_i^e \, d\omega \approx \sum_{\Omega} g_i^0 \cdot g_i^e \quad (20)$$

$$d_g = d(g_i^0, g_i^e) = \sqrt{g_i^0, -g_i^e, g_i^0, -g_i^e} = \sqrt{\sum_{\Omega} (g_i^0 - g_i^e)^2} \quad (21)$$

$$p_{cont} = 1 - e^{-d_g} \quad (22)$$

Then, to enable the calculation of probabilities p_{cont} and p_{cont} , we have to model the probability distribution of the stochastic process ϕ_i within the interval B_s .

We are now defining the operator $\mathfrak{C}(\cdot, \cdot)$ within the universe Ω (23). Basically, this operator indicates the number of elements in the universe that are between two provided values; that is, $card\{\cdot\}$ the standard cardinality operator. This operator is coherent as Ω is a subset of the field of the real number where a strict order relation is defined. This operator is a positive operator as the target set is the set of natural numbers $\mathbb{N} \cup \{0\}$.

$$\begin{aligned} \mathfrak{C} : \Omega \times \Omega &\rightarrow \mathbb{N} \cup \{0\} \\ \mathfrak{C}(\omega_1, \omega_2) &= card\{\omega_k : \omega_1 \leq \omega_k < \omega_2\} \end{aligned} \quad (23)$$

Now, we are assuming the stochastic process ϕ_i is also locally ergodic in B_s . Thus, and according to the Birkhoff ergodic theorem, the additions A_m of the composite function $\mathfrak{C} \circ \phi_i$ (restricted to B_s , $\phi_i|_{B_s}$) converge "almost surely" to the statistical expected value of the composite function $\mathfrak{C} \circ I_{\Omega}$ (24), where I_{Ω} is the identity function in the universe Ω .

$$\begin{aligned} A_m &= \sum_{r=0}^m \left(\mathfrak{C} \circ (\phi_i|_{B_s})^m \right) \\ \frac{A_m}{m} &\rightarrow \mathbb{E}[\mathfrak{C} \circ I_{\Omega}] \end{aligned} \quad (24)$$

Now, we are considering a partition Π_{Ω} of the universe Ω , composed of Δ different subsets π_i (25). All subsets π_i have the same measure ℓ_i , understood as the Lebesgue measure (26).

$$\begin{aligned} \Pi_{\Omega} &= \{\pi_i \mid i = 1, \dots, \Delta\} \\ \Pi_{\Omega} &= \bigcup_{i=1}^{\Delta} \pi_i \end{aligned} \quad (25)$$

$$\begin{aligned} \ell_i &= \ell(\pi_i) = b_i - a_i \\ \text{being } \pi_i &= (a_i, b_i) \subset B_s \subset \mathbb{R} \end{aligned} \quad (26)$$

If we restrict the previous operator $\mathfrak{C}(\cdot, \cdot)$ to any subset π_i , $\mathfrak{C}|_{\pi_i}$ and evaluate this operator in the limits (a_i, b_i) of this set π_i , the statistical expected value of the composite

function $\mathfrak{C} \mid \pi_i \circ I_{\pi_i}$ is “almost surely” identical to the expression for the Laplace rule employed to calculate the probability of an event (27).

$$\begin{aligned} \frac{A_m}{m} &= \frac{1}{m} \sum_{r=0}^m \left(\mathfrak{C} \mid \pi_i \circ \left(\phi_i \mid_{B_s, \pi_i} \right)^m \right) = \\ &= \frac{1}{m} \text{card} \{ x_i[n] : n \in B_s \wedge a_i \leq x_i[n] < b_i \} \rightarrow \mathbb{E} \left[\mathfrak{C} \mid \pi_i \circ I_{\pi_i} \right] \end{aligned} \quad (27)$$

In this case, the event under study is the fact a sensor node generates a sample belonging to π_i in the time interval B_s . In conclusion, we are studying the probability distribution of the stochastic process ϕ_i in the interval B_s .

We are now defining a function $f(\cdot)$ associating the mean point σ_i of every subset π_i with the additions A_m (28). In other words, through the additions A_m , we are generating a discrete probability function $f(\cdot)$, which “almost surely” converges to the actual probability distribution of the stochastic process ϕ_i in the interval B_s and the points σ_i (29).

$$f(\sigma_i) = f\left(\frac{\ell_i}{2}\right) = \frac{A_m}{m} = \frac{1}{m} \text{card} \{ x_i[n] : n \in B_s \wedge a_i \leq x_i[n] < b_i \} \quad (28)$$

$$f(\sigma_i) \rightarrow \phi_i \mid_{B_s, \pi_i}(n_0, \sigma_i) \quad (29)$$

To calculate the probability distributions g_i^0 and g_i^e , the same process as described before may be employed, but considering the proper time interval and a new universe Σ composed by points σ_i (30). Probability p_{rx} can be directly obtained using function $f(\sigma_i)$.

$$\Sigma = \{ \sigma_i \mid i = 1, \dots, \Delta \} \quad (30)$$

3.2. Candidates to Curated Time Series: Calculation

The first step to improve the quality of the time series $x_i[n]$ produced by sensor nodes S_i in Industry 4.0 systems is to find the candidate series \mathfrak{X} to be the curated flow we are looking for. Initially, any series \mathfrak{X}_c could be a candidate, and probability p_i^* will be the indicator to select the final curated series $x_i^*[n]$. However, this approach is almost impossible to implement in practice, as the universe of time series in the curation interval $[n_1, n_2]$ is infinite. Moreover, as this is not a free mathematical problem, some physical restrictions inherit from the Industry 4.0 system we are modeling must be considered.

First, sensor nodes have an operational range $[x_{min}, x_{max}]$, which introduces a hard restriction: no candidate \mathfrak{X}_c with samples in the exterior of the interval $[x_{min}, x_{max}]$ is in the final curated series $x_i^*[n]$ (31).

$$\mathfrak{X}_c[n] \notin [x_{min}, x_{max}] \mid n \in [n_1^e, n_2^e] \Rightarrow \mathfrak{X}_c[n] \neq x_i^*[n] \mid n \in [n_1^e, n_2^e] \quad (31)$$

Second, Industry 4.0 systems monitor physical processes, which are continuous and smooth (as natural variables), so no gaps or abrupt changes may appear in the curated time series. In this context, curated series $x_i^*[n]$ in the interval $[n_1, n_2]$ must be continuous and coherent with time series in the surrounds of this interval, i.e., in the expanded time interval $[n_1^e, n_2^e]$. In that way, analytic function describing the evolution of the Industry 4.0 system in the curation interval $[n_1, n_2]$ must also be able to describe the system evolution in the expanded interval $[n_1^e, n_2^e]$. To apply this restriction, the best way to calculate the candidates \mathfrak{X}_c is using interpolation techniques.

Different interpolation techniques may generate different candidates \mathfrak{X}_c , so in this work, we are considering the most powerful, popular and well-behaved interpolation solutions: Newton’s divided differences, Hermite interpolation, splines, Taylor interpolation and Lagrange polynomial.

These techniques are evaluated using the E points $x_{ext}^i[n]$ which belong to the expanded interval $[n_1^e, n_2^e]$, but they are not included in the curation interval $[n_1, n_2]$ (as

data in the curation interval may be wrong and introduce false information in our algorithm) (32).

$$x_{ext}^i[n] = \left\{ x_{ext}^i[n_j^{ext}] \mid j = 1, \dots, E \right\} = x_i[n] : n \in \{[n_1^e, n_2^e] \cap [n_1, n_2]\} = \{n_j^{ext} \mid j = 1, \dots, E\} \quad (32)$$

Candidate $\mathfrak{X}_1[n]$ is obtained using the Newton's divided differences technique. In this case, as the independent variable is the discrete time n , traditional expressions for Newton's interpolation are slightly modified. Specifically, given the E points in $x_{ext}^i[n]$, candidate $\mathfrak{X}_1[n]$ is a polynomial with order $E - 1$ (33). Coefficients (named as divided differences) may be easily calculated using simple mathematical operations, which reduces the computational time, enabling a real-time operation (34).

$$\mathfrak{X}_1[n] = \delta_0 + \sum_{z=1}^E \delta_z \cdot \prod_{r=1}^z (n - n_r^{ext}) \quad (33)$$

$$\begin{aligned} \delta_0 &= x_{ext}^i[n_1^{ext}] \\ \delta_1 &= \frac{x_{ext}^i[n_2^{ext}] - \delta_0}{n_2^{ext} - n_1^{ext}} \\ \delta_2 &= \frac{\frac{x_{ext}^i[n_3^{ext}] - \delta_0}{n_3^{ext} - n_2^{ext}} - \delta_1}{n_3^{ext} - n_1^{ext}} \\ \delta_3 &= \frac{\frac{\frac{x_{ext}^i[n_4^{ext}] - \delta_0}{n_4^{ext} - n_3^{ext}} - \delta_1}{n_4^{ext} - n_2^{ext}} - \delta_2}{n_4^{ext} - n_1^{ext}} \\ &\dots \end{aligned} \quad (34)$$

Candidate $\mathfrak{X}_2[n]$ is calculated through the Lagrange polynomial interpolation algorithm. In this case, the candidate is just a linear combination of data in sequence $x_{ext}^i[n]$ (35). Besides, in this case, the order β of the interpolation polynomial may be selected (if it is lower than E , number of points in $x_{ext}^i[n]$). In general, polynomial with orders above six are not suitable (because they present unnatural fluctuations), but this parameter is free to be selected according to the Industry 4.0 system under study.

$$\mathfrak{X}_2[n] = \sum_{z=1}^{\beta} x_{ext}^i[n_z^{ext}] \prod_{r=1, r \neq z}^{\beta} \frac{n - n_r^{ext}}{n_z^{ext} - n_r^{ext}} \quad \beta < E \quad (35)$$

The third candidate $\mathfrak{X}_3[n]$ is obtained using the Hermite interpolation theory. In this case, besides the sequence $x_{ext}^i[n]$, it is also necessary to know the value of the first order derivative $\dot{x}_{ext}^i[n]$ in the points n_j^{ext} . When managing discrete-time sequences, this may be easily calculated using first-order finite differences. In general, we are using a central difference (36), as it presents a much lower error. However, if either time point n_{j-1}^{ext} or time point n_{j+1}^{ext} do not exist, we can employ the forward difference (37) or the backward difference (38) respectively (and although a higher numerical error is introduced).

$$\dot{x}_{ext}^i[n_j^{ext}] = \frac{x_{ext}^i[n_{j+1}^{ext}] - x_{ext}^i[n_{j-1}^{ext}]}{n_{j+1}^{ext} - n_{j-1}^{ext}} \quad (36)$$

$$\dot{x}_{ext}^i[n_j^{ext}] = \frac{x_{ext}^i[n_{j+1}^{ext}] - x_{ext}^i[n_j^{ext}]}{n_{j+1}^{ext} - n_j^{ext}} \quad j < 1 \quad (37)$$

$$x_{ext}^i[n_j^{ext}] = \frac{x_{ext}^i[n_j^{ext}] - x_{ext}^i[n_{j-1}^{ext}]}{n_j^{ext} - n_{j-1}^{ext}} \quad j > E \quad (38)$$

If both time instants n_{j-1}^{ext} and n_{j+1}^{ext} do not exist, the first-order derivative cannot be calculated for instant n_j^{ext} . In that case, that point n_j^{ext} is not considered to calculate the candidate sequence $\mathfrak{X}_3[n]$.

Given the Lagrange polynomial $\mathcal{L}_r[n]$ (39), and its first order derivative $\dot{\mathcal{L}}_r[n]$, the Hermite interpolated sequence may be calculated through an osculating polynomial (40). This approach generates high-quality candidates, which may integrate large amount of points with a reduced computational cost.

$$\mathcal{L}_r[n] = \prod_{z=1, z \neq r}^E \frac{n - n_z^{ext}}{n_r^{ext} - n_z^{ext}} \quad (39)$$

$$\begin{aligned} \mathfrak{X}_3[n] &= \sum_{r=1}^E x_{ext}^i[n_r^{ext}] \cdot \mathcal{H}_r[n] + \sum_{r=1}^E x_{ext}^i[n_r^{ext}] \cdot \dot{\mathcal{H}}_r[n] \\ \mathcal{H}_r[n] &= \left(1 - 2(n - n_r^{ext}) \dot{\mathcal{L}}_r[n]\right) \cdot \mathcal{L}_r^2[n] \\ \dot{\mathcal{H}}_r[n] &= (n - n_r^{ext}) \cdot \mathcal{L}_r^2[n] \end{aligned} \quad (40)$$

Candidate $\mathfrak{X}_4[n]$ is based on Taylor's interpolation. Formally, this approach only requires one sample at the time instant n_{taylor}^{ext} , so it is a very good candidate for the initial moments of the Industry 4.0 system operation, when collected data are very limited. In practice, however, this method requires the use of different successive r -th derivatives $\overbrace{x_{ext}^i}^{(z)}[n]$. They can be easily obtained using the central, forward or backward differences we already described (36)–(38), but this needs some additional samples. In this approach, the order β of the interpolation polynomial can be also selected. Therefore, in general, for a given order β , this method needs between $\beta + 1$ and $\beta + 2$ samples. As said before, polynomial with orders above six show some unnatural variations. On the other hand, for very low values of β , the numerical error is also high. A balance between both factors must be reached.

In this context, candidate $\mathfrak{X}_4[n]$ may be easily obtained (41).

$$\mathfrak{X}_4[n] = \sum_{z=1}^{\beta} \frac{\overbrace{x_{ext}^i}^{(z)}[n_{taylor}^{ext}]}{z!} (n - n_{taylor}^{ext})^z \quad \beta < E \quad (41)$$

Finally, candidate $\mathfrak{X}_5[n]$ is obtained through splines. Using the splines technique, candidate is just a segmented polynomial (42).

$$\mathfrak{X}_5[n] = \begin{cases} \mathfrak{X}_5^1[n] & n \in [n_1^{ext}, n_2^{ext}) \\ \dots & \dots \\ \mathfrak{X}_5^j[n] & n \in [n_j^{ext}, n_{j+1}^{ext}) \\ \dots & \dots \\ \mathfrak{X}_5^{E-1}[n] & n \in [n_{E-1}^{ext}, n_E^{ext}) \end{cases} \quad (42)$$

This polynomial may have different orders (from one to three), but it is well-known that cubic splines is the solution generating the best candidates [53] (they are smooth, contrary to linear splines, and they adapt to a larger range of system behaviors). For each pair of time instants n_j^{ext}, n_{j+1}^{ext} a new cubic polynomial $\mathfrak{X}_4^j[n]$ is defined (43). In this polynomial $\mathfrak{X}_5^j[n]$, variables μ_j are unknown and are calculated through the continuity and

smoothness conditions (44). In this proposal, we are using natural splines, so for any point n_j^{ext} where it is impossible to calculate the first-order derivative $\dot{\mathcal{X}}_5^j[n]$ or the second-order derivative $\ddot{\mathcal{X}}_5^j[n]$, these values are considered null (zero).

$$\begin{aligned} \mathcal{X}_5^j[n] &= \frac{\mu_j}{6(n_{j+1}^{ext} - n_j^{ext})} (n_{j+1}^{ext} - n)^3 + \frac{\mu_{j+1}}{6(n_{j+1}^{ext} - n_j^{ext})} (n - n_j^{ext})^3 \\ &+ \left(\frac{x_{ext}^j[n_{j+1}^{ext}]}{n_{j+1}^{ext} - n_j^{ext}} + \frac{\mu_{j+1}(n_{j+1}^{ext} - n_j^{ext})}{6} \right) \cdot (n - n_j^{ext}) \\ &+ \left(\frac{x_{ext}^j[n_j^{ext}]}{n_{j+1}^{ext} - n_j^{ext}} - \frac{\mu_j(n_{j+1}^{ext} - n_j^{ext})}{6} \right) \cdot (n_{j+1}^{ext} - n) \end{aligned} \quad (43)$$

$$\begin{aligned} \mathcal{X}_5^j[n_j^{ext}] &= \mathcal{X}_5^{j+1}[n_j^{ext}] \quad j = 1, \dots, E-1 \\ \dot{\mathcal{X}}_5^j[n_j^{ext}] &= \dot{\mathcal{X}}_5^{j+1}[n_j^{ext}] \quad j = 1, \dots, E-1 \\ \ddot{\mathcal{X}}_5^j[n_j^{ext}] &= \ddot{\mathcal{X}}_5^{j+1}[n_j^{ext}] \quad j = 1, \dots, E-1 \end{aligned} \quad (44)$$

This final candidate is more computationally costly to calculate, as we are introducing a system of linear equations that must be solved to obtain the final expression for the candidate. However, this method creates high-quality, curated data series (with a reduced error), and (currently) linear equations are easily solved using numerical methods (especially in strong cloud infrastructures or Industry 4.0 software platforms).

3.3. Malfunction Modeling

The calculation of probabilities p_i and p_{mal} is directly associated to functions λ_l , which model the data malfunctions in the Industry 4.0 platform. In this proposal, four different malfunctions are addressed: jitter (including inactivity periods in hardware nodes), communication delays, numerical errors in microprocessors and electromagnetic interferences (including data losses and transmissions errors). As said before, to calculate p_i , we must consider the probability of all these functions λ_l to be the identity function I_Ω (null effect), while probability p_{mal} is obtained considering (for each candidate) the situation when $\lambda_l(\mathcal{X}_c[n]) = x_i[n]$.

Jitter $\mathcal{J}(\xi)$ is probably the most harmful malfunction among all the ones considered in this proposal. Jitter is the maximum fluctuation in the communication delays, transmission periods or clock synchronization that causes samples $x_i[n]$ to be randomly ordered compared to the original ones in $y_i[n]$ (45). Jitter is represented by a function $\tau_{jitter}[k]$ taking values in the interval $[0, \xi_k]$, where ξ_k is a realization (different for each value of k) of a random variable $\mathcal{J}(\xi)$ taking values in a continuous universe: the field of positive real numbers (46). Because of jitter, no samples may be received for long periods, while in other moments, large amounts of samples may be received and interfere.

$$x_i[n] = \lambda_1(y_i[n]) = \sum_{\forall k : n=k-\tau_{jitter}[k]} y_i[k - \tau_{jitter}[k]] \quad (45)$$

$$\begin{aligned} \tau_{jitter}[k] &\in [0, \xi_k] \\ P(\xi_k) &= \mathcal{J}(\xi_k) \quad \xi_k \in \mathbb{R} \cup \{0\} \end{aligned} \quad (46)$$

Total jitter $\mathcal{J}(\xi)$ in Industry 4.0 systems is, actually, the composition of two different and independent sources: random jitter $\mathcal{J}_{random}(\xi)$ and deterministic jitter $\mathcal{J}_{deter}(\xi)$ (47). Operator $*$ represents the convolution.

Deterministic jitter is caused by three different additive effects (well-known and modeled through deterministic expressions): fluctuations in the clock periods and edges, variations in the data packet length and sleep periods in the sensor nodes or the communi-

cation channels [54]. Although these three effects are deterministic, they are also dependent on random variables, such as the clock-duty cycle and frequency, the packet length, and the duration of the blocking, respectively. In conclusion, deterministic jitter $\mathcal{J}_{deter}(\xi)$ is a Gaussian distribution, according to the central limit theorem (48) with mean value m_{det} and standard deviation s_{det} .

$$\mathcal{J}(\xi) = \mathcal{J}_{deter}(\xi) * \mathcal{J}_{random}(\xi) \quad (47)$$

$$\mathcal{J}_{deter}(\xi) = \frac{1}{s_{det}\sqrt{2\pi}} e^{-\frac{(\xi-m_{det})^2}{2s_{det}^2}} \quad (48)$$

On the other hand, many other random and uncontrolled effects, such as thermal oscillations, flicker or shot noise, may result in levels of jitter that cannot be predicted or calculated in any way. This is known as random jitter $\mathcal{J}_{random}(\xi)$ and, according to the central limit theorem, is also a Gaussian distribution (49) with mean value m_{ran} and standard deviation s_{ran} .

$$\mathcal{J}_{random}(\xi) = \frac{1}{s_{ran}\sqrt{2\pi}} e^{-\frac{(\xi-m_{ran})^2}{2s_{ran}^2}} \quad (49)$$

The convolution of these two Gaussian distributions is a third Gaussian distribution (50). The mean value m and the standard deviation s depend on the scenario and Industry 4.0 system, but most modern 5G solutions establish the mean value m around 1 millisecond and the standard deviation s is one magnitude order lower. These values are aligned with the expected performance for ultra-reliable low latency communications (URLLC) in 5G networks and scenarios [55].

$$\mathcal{J}(\xi) = \frac{1}{\sqrt{2\pi} \sqrt{s_{ran}^2 + s_{det}^2}} e^{-\frac{(\xi-m_{ran}-m_{det})^2}{2(s_{ran}^2 + s_{det}^2)}} = \frac{1}{s\sqrt{2\pi}} e^{-\frac{(\xi-m)^2}{2s^2}} \quad (50)$$

Communication delays $\mathcal{D}(\xi)$ are not as harmful as jitter as they are a linear transformation (51), but they may cause delayed decisions and other similar problems. As all malfunctions, delays are a random effect, and they are described by a random variable taking values in a continuous universe: the field of positive real numbers (51).

In this case, we are decomposing the total delay $\mathcal{D}(\xi)$ in three different contributions (52): delay in the output queue (sensor node) $\mathcal{D}_{out}(\xi)$, transmission delay $\mathcal{D}_{tran}(\xi)$, and delay in the input queue (software module) $\mathcal{D}_{in}(\xi)$.

$$\begin{aligned} x_i[n] &= \lambda_2(y_i[n]) = y_i[n - \xi] \\ P(\xi) &= \mathcal{D}(\xi) \quad \xi \in \mathbb{R} \cup \{0\} \end{aligned} \quad (51)$$

$$\mathcal{D}(\xi) = \mathcal{D}_{out}(\xi) + \mathcal{D}_{tran}(\xi) + \mathcal{D}_{in}(\xi) \quad (52)$$

Both delays associated with queues are formally identical. The traffic theory allows obtaining the probability distribution for both components $\mathcal{D}_{out}(\xi)$ and $\mathcal{D}_{in}(\xi)$. In both cases, we are using a Poisson model M/M/1/P (in Kendall notation), where the sample generation rate ψ and the serving rate η follow a Poisson distribution, Ψ is the mean sample generation rate, Θ is the mean serving time and Γ is the time period taken as reference (typically one hour). P is the number of samples allowed in the system (53).

If we assume a FIFO (first in, first out) managing strategy for both queues in our model, we can define a Markov chain for describing the queues state. In that situation, the traffic theory and statistical laws define the queueing delay as an exponential distribution (54).

$$\begin{aligned} \psi(z) &= \frac{(\Psi \cdot \Gamma)^z}{z!} \exp(-\Psi \cdot \Gamma) \\ \eta(z) &= \frac{(\Theta \cdot \Gamma)^z}{z!} \exp(-\Theta \cdot \Gamma) \end{aligned} \quad (53)$$

$$\mathcal{D}_{out}(\xi) \sim \mathcal{D}_{in}(\xi) = \left(\frac{\psi}{\eta}\right)^{\eta \cdot \xi} \frac{1 - \frac{\psi}{\eta}}{1 - \left(\frac{\psi}{\eta}\right)^{P+1}} \quad (54)$$

Regarding the transmission delay $\mathcal{D}_{tran}(\xi)$, several different random and unknown variables affect the final value: data packet length, channel capacity, physical configuration of the scenario, etc. Therefore, and considering the central limit theorem, the probability distribution of the transmission delay is a Gaussian function with mean value m_d and standard deviation s_d (55).

$$\mathcal{D}_{tran}(\xi) = \frac{1}{s_d \sqrt{2\pi}} e^{-\frac{(\xi - m_d)^2}{2s_d^2}} \quad (55)$$

Numerical errors in microprocessors are caused by the limited precision of hardware components. Basically, these errors are associated with two different modules: the analog-to-digital converter (ADC) and the arithmetic combinational modules. In the ADC, because of the quantification process, samples are irreversibly modified. Because of the arithmetic combinational modules, operations with large numbers may be truncated to the maximum number admissible in the microprocessor, ρ_{max} .

In any case, the numerical error $\mathcal{N}(\xi)$ is an additive effect (56), composed of two different sub-effects: the ADC error $\mathcal{N}_{ADC}(\xi)$ and the arithmetic error $\mathcal{N}_{ari}(\xi)$ (57). In this case, random variable $\mathcal{N}(\xi)$ takes values in the universe of real numbers (56).

$$\begin{aligned} x_i[n] &= \lambda_3(y_i[n]) = y_i[n] + \xi \\ P(\xi) &= \mathcal{N}(\xi) \quad \xi \in \mathbb{R} \end{aligned} \quad (56)$$

$$\mathcal{N}(\xi) = \mathcal{N}_{ari}(\xi) + \mathcal{N}_{ADC}(\xi) \quad (57)$$

Regarding the quantification error in the ADC, we are assuming the sensor node is configured according to the manufacturer's restrictions, and the analog signal $y_i(t)$ being digitalized is limited in amplitude to the operation range $[-Y_{max}^{ADC}, Y_{max}^{ADC}]$ of the ADC device. In this context, given an ADC device with u intervals with an amplitude of Σ units (58), the error will be restricted to the interval $\left[-\frac{\Sigma}{2}, \frac{\Sigma}{2}\right]$ (the maximum error appears when the sample is exactly in the middle of an ADC interval) (59). All values within the proposed interval have the same probability, so the distribution is uniform (60).

$$\Sigma = \frac{Y_{max}^{ADC}}{u} \quad (58)$$

$$\mathcal{N}_{ADC}(\xi) : \xi \in \left[-\frac{\Sigma}{2}, \frac{\Sigma}{2}\right] \quad (59)$$

$$\mathcal{N}_{ADC}(\xi) = \frac{1}{\Sigma} \quad \forall \xi \quad (60)$$

On the other hand, errors caused by the limited precision of arithmetic devices, $\mathcal{N}_{ari}(\xi)$, may take any value as there is no superior limit. However, all these values do not have the same probability, as lower errors are much more probable than higher errors (input parameters are also limited and system designers usually also consider this problem in their code). In particular, we are proposing the arithmetic error follows an exponential distribution (61). If we assume the maximum number the microprocessor can represent is ρ_{max} , the worst situation to happen (where arithmetic error is highest) is the addition of two parameters with this maximum value ρ_{max} . The result, then, presents a maximum

error of ρ_{max} units. Thus, the mean value for our exponential distribution is ρ_{max} (and the distribution gets totally defined).

$$\mathcal{N}_{ari}(\xi) = \frac{1}{\rho_{max}} \cdot e^{-\frac{\xi}{\rho_{max}}} \quad (61)$$

Finally, electromagnetic interferences are responsible for two main problems: sample losses and data corruption. At the data level, electromagnetic interferences show as a bit error rate (BER). If, because of BER, the number of corrupted bits in a sample $x_i[n]$ is above a limit γ_{lim} , the sample is corrupted and is rejected and not received. If the number of corrupted bits is below that limit γ_{lim} , the sample $x_i[n]$ is corrupted by an additive effect but is still received (62). Hereinafter, we are considering Λ as the length (in bits) of samples. This is a random variable with a uniform distribution in the interval $[0, \Lambda_{max}]$.

$$x_i[n] = \lambda_4(y_i[n]) = \begin{cases} \text{null} & \text{if } BER \cdot \Lambda > \gamma_{lim} \\ x_i[n] + \xi & \text{if } BER \cdot \Lambda < \gamma_{lim} \end{cases} \quad (62)$$

$$P(\Lambda) = \frac{1}{\Lambda_{max}} \quad (63)$$

The calculation of the additive distortion ξ is also based on the length of samples Λ (in bits). The error, then, must be included in the interval $[-(2^\Lambda - 1), 2^\Lambda - 1]$, the maximum value that can be represented using Λ bits. Errors have the same probability in all bits, so the probability distribution of distortion values ξ is uniform.

$$P(\xi) = \frac{1}{2^{\Lambda+1}} \quad (64)$$

$$\xi \in [-(2^\Lambda - 1), 2^\Lambda - 1]$$

Finally, we must calculate the value of BER using the level of electromagnetic interferences. Given the signal power per bit E_b and the power of interferences N_0 , the signal theory establishes the BER is obtained through the complementary error function $erfc(\cdot)$ (65).

$$BER = \frac{1}{2} erfc\left(\sqrt{\frac{E_b}{N_0}}\right) = \frac{2}{\sqrt{\pi}} \int_{\sqrt{\frac{E_b}{N_0}}}^{\infty} e^{-t^2} dt \quad (65)$$

3.4. Final Data Generation and Correction Step

After calculating probabilities p_i^* and p_i and selecting the candidate $\mathcal{X}_c[n]$ associated to the higher probability p_i^* among all five calculated candidates, different situations may occur. First, if offline data curation is being performed, no new information is expected after the performed calculations. Then, results are final. However, three possibilities are considered:

- If probability p_i^* is clearly higher than p_i , candidate $\mathcal{X}_c[n]$ associated to this probability p_i^* is selected as the curated data series $x_i^*[n]$ and initially received information $x_i[n]$ is deleted. In this proposal, we are considering a difference higher than 10% (66) as the limit to take this action.

$$\frac{p_i^*}{p_i} \geq 1.1 \quad (66)$$

- On the contrary, if probability p_i is clearly higher than probability p_i^* (67), all candidates are rejected and initially received data flow $x_i[n]$ is accepted as the curated data series $x_i^*[n]$.

$$\frac{p_i}{p_i^*} \geq 1.1 \quad (67)$$

- If neither probability p_i^* nor p_i is clearly higher than the other (68), the final curate data series $x_i^*[n]$ cannot be concluded to be the candidate $\mathcal{X}_c[n]$ or the originally received

data $x_i[n]$. Thus, the curated data series $x_i^*[n]$ is obtained as the arithmetic average of both series (69).

$$0.9 < \frac{p_i}{p_i^*} < 1.1 \quad (68)$$

$$x_i^*[n] = \frac{1}{2}(x_i[n] + \mathfrak{X}_c[n]) \quad n \in [n_1, n_2] \quad (69)$$

Second, if real-time data curation is being performed, there will be a future time instant n_{fut} , belonging to the expanded interval $[n_1^e, n_2^e]$, whose associated sample $x_i[n_{fut}]$ is not available at the time instant n_0 when the curated series is obtained, but it is received in the future. This new information may affect the previous calculation $x_i^{*old}[n]$, then a correction phase is considered to update the previous results.

First, the new sample $x_i[n_{fut}]$ is compared to the previously obtained curated sample $x_i^{*old}[n_{fut}]$. Then, if the difference is high enough (above 10%) (70), all the curation process is redone considering the new information.

$$\frac{x_i[n_{fut}]}{x_i^{*old}[n_{fut}]} \geq 1.1 \vee \frac{x_i[n_{fut}]}{x_i^{*old}[n_{fut}]} \leq 0.9 \quad (70)$$

However, if the difference between samples $x_i[n_{fut}]$ and $x_i^{*old}[n_{fut}]$ is not relevant, to reduce the global computational cost of the proposed solution, the corrected curated data series $x_i^*[n_{fut}]$ is obtained as the arithmetic average of old curated sample $x_i^{*old}[n_{fut}]$ and the new information $x_i[n_{fut}]$ (71).

$$x_i^*[n_{fut}] = \frac{1}{2}(x_i^{*old}[n_{fut}] + x_i[n_{fut}]) \quad (71)$$

Table 1 summarizes the most relevant symbols and variables considered in the proposed predictor-corrector scheme.

Table 1. Most relevant symbols and variables.

Parameter	Meaning	Parameter	Meaning
\mathfrak{T}	Industry 4.0 platform	\mathcal{S}_i	Sensor node
N	Number of data flows	λ_l	Malfunction function
L	Number of malfunctions	$x_i[n]$	Original data series
Ω	Universe of values for data series	f_s^i	Sampling frequency
n_0	Current time instant	n_1^e	Limit in the expanded time interval
$x_i^*[n]$	Curated time series	p_{mal}	Probability of malfunction
\mathfrak{X}_c	Candidate to curated data series	d_g	Distance between distributions
p_{rx}	Probability of data in reception	$y_i[n]$	Real data generated by physical sensors
C	Number of candidates	p_i	Probability of original data
I_Ω	Identity function	B_s	Stationary time interval
Π_Ω	Partition of the universe	Δ	Number of elements in a partition
	Ω		

4. Experimental Validation and Results

To evaluate the performance and usability of the proposed technology, an experimental validation was designed and carried out. In this experimental phase, we analyzed the precision of the proposed prediction-correction method, but we also evaluated its behavior in terms of scalability and required computational time. This section describes the proposed experimental methods and the obtained results.

4.1. Experiments: Methods and Materials

Four different experiments were planned and developed. Three of them were based on simulation techniques, while the fourth one was supported by a real hardware infrastructure.

The first experiment evaluated the evolution of the computational cost of the proposed solution when deployed in an Industry 4.0 system. To do that, the number of sensor nodes N in the scenario was varied, and the total computational time required to curate all data series in real time was monitored. This experiment was focused on the solution's cost and its scalability. Time measurements and results were captured and displayed as relative values normalized by the sampling period T_i . In that way, when processing time was above the sampling period T_i , we determined the proposed framework was congested (buffers will grow until the entire software module becomes unavailable). In this experiment, MATLAB 2019B software was employed to build a simulation scenario where we could change the number of nodes N in an easy way. The experiment was repeated for different values of the sampling period T_i .

The second experiment was also focused on analyzing the computational cost of the proposed solution and its scalability, but in this case when performing an offline data curation. In this case, the relative length of the expanded time interval $[n_1^e, n_2^e]$, with respect to the curation interval $[n_1, n_2]$, (72), was varied in different experiment's realizations, and the total computational time required to curate all the samples in the curation interval was monitored.

$$\frac{n_2^e - n_1^e}{n_2 - n_1} \quad (72)$$

As in the first experiment, to reduce the external validity threats to this experiment and make results more general, computational time was expressed as a relative number respect to the sampling period T_i . This experiment was also based on simulation techniques using the MATLAB 2019B software, where an Industry 4.0 system was built and run. All sensors were generating data every 30 s. The experiment was repeated for different numbers of sensor nodes in the Industry 4.0 system, N .

The third experiment aimed to evaluate the precision and curation success of the proposed predictive-correction algorithm. In this case, the experiment was also based on simulation techniques using the MATLAB 2019B software. Measures about the proposal's precision were obtained as the mean square error, MSE, (73) for the entire curated data flow and all the sensor nodes in the scenario. This error basically evaluated how different the obtained curated data series $x_i^*[n]$ was from the original information generated by sensor nodes $y_i[n]$. To improve the clarity in the results, this MSE was expressed as a percentage (74). The same experiment was performed for the two proposed operation modes: offline data curation and real-time data curation. Furthermore, the proposed experiment was repeated for different relative lengths of the expanded time interval $[n_1^e, n_2^e]$, with respect to the curation interval $[n_1, n_2]$ (72). All sensors were generating data every 30 s.

$$MSE = \frac{1}{N \cdot (n_2 - n_1)} \sum_{i=1}^N \sum_{n=n_1}^{n_2} (x_i^*[n] - y_i[n])^2 \quad (73)$$

$$MSE (\%) = \frac{MSE}{\frac{1}{N \cdot (n_2 - n_1)} \sum_{i=1}^N \sum_{n=n_1}^{n_2} (y_i[n])^2} \cdot 100 \quad (74)$$

Furthermore, to evaluate the improvement provided by the proposed scheme compared to existing mechanisms, results in the third experiment were compared to results reported by state-of-the-art solutions [56]. Specifically, to perform a coherent and valid comparison, we selected a low-cost, error-correction framework for wireless sensor networks. This scenario was similar to Industry 4.0 applications, and the selected solution also included a prediction and a correction phase, so the mechanisms were technically comparable, so results could be also compared. Results for this state-of-the-art solution [56] were obtained through a secondary simulation scenario with the same characteristics and setup employed for the proposed new approach.

For these three initial experiments, a simulation scenario was built and run using the MATLAB 2019B suite. The simulation scenario represented a chemical industry where environmental data are collected to make decisions about how safe it is to work on the pending activities. Four different kinds of sensor nodes were considered: temperature sensors, humidity sensors, CO₂ sensors (air quality) and volatile compounds sensors (detecting dangerous gas emissions). The scenario included a random composition, using all these four types of sensor nodes. All sensors generated data every T_i seconds, although they were not synchronized. Each sensor started operating at a random instant within the first minute of the system running. Simulated sensor nodes were designed to represent an ESP-32 microprocessor (with a 12-bit ADC and a 16-bit architecture). Figure 2 shows the experimental simulation setup.

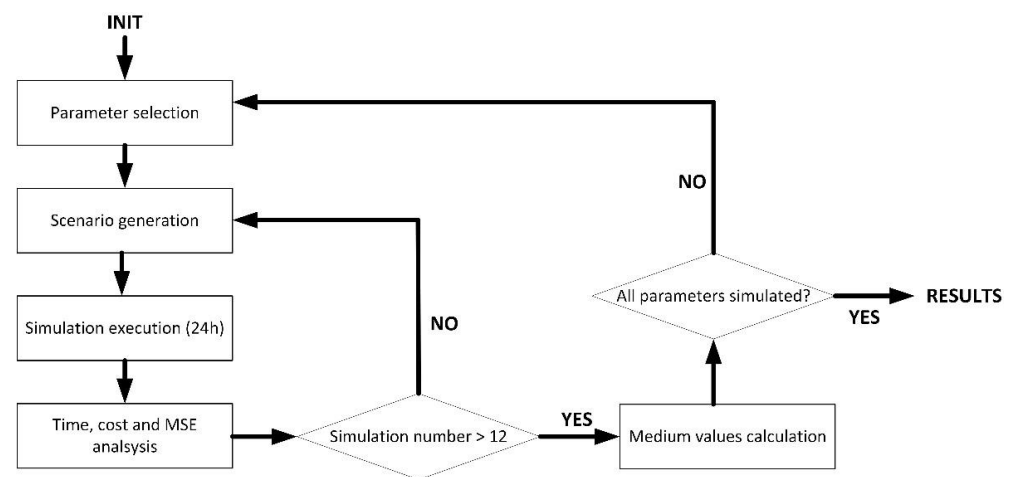


Figure 2. Experimental simulation setup.

The simulated scenario was based on a minimum-sized industry, where distances were never larger than 50 m. The decision-making software module collected information from sensor nodes using LoRaWAN wireless technology. The global environment was suburban, so the level of interferences was moderate-low.

Malfunctions were represented by models and functions included in the MATLAB libraries, so we guaranteed the independence of the system configuration phase and the system evaluation phase. In that way, results were more relevant. Models for jitter were introduced from Simulink and included duty-cycle-distortion deterministic jitter, random jitter, sinusoidal jitter and noise. In this work, only deterministic and random jitter were considered. The different types of jitter were injected into devices according to the IBIS-AMI specifications [57]. Delays were managed through the Control System Toolbox, which allowed integrating the input delay, the output delay and the independent transport delays in the transfer function and the frequency-response data. In this work, we are using first-order plus dead time models and state-space models with input and output delays. Regarding BER, MATLAB includes the Bit Error Rate Analysis App Environment, which can integrate different instances of the numerical models generated through a Monte Carlo Analysis Simulink block and simulation. Finally, numerical errors were introduced and modeled as numerical noise in devices, using the Simulink framework and the IBIS-AMI specifications (in a similar way as described and conducted for jitter models).

Simulations represented a total of 24 h of system operation. To remove the impact of exogenous variables in the results, each simulation was repeated 12 times. Results were calculated as the mean value of all these simulations for each system configuration. All simulations were performed using a Linux architecture (Ubuntu 20.04 LTS) with the following hardware characteristics: Dell R540 Rack 2U, 96 GB RAM, two processors (Intel Xeon Silver 4114 2.2 GB, HD 2 TB SATA 7.2K rpm). Simulations were performed under isolation conditions: Only the MATLAB 2019B suite was installed and running in the

machine; all other services, software or internet connection were stopped or removed. The objective was to remove as much as possible all validity threats. The global simulation time was variable and automatically calculated by the system to get data representing 24 h of system operation.

The fourth and final experiment also aimed to evaluate the precision and curation success of the proposed predictive-correction algorithm. However, in the last experiment, we employed a real hardware platform. In this case, in an emulation environment representing the referred chemical industry, four nodes (one of each type) were deployed together with a LoRaWAN gateway. The sensor nodes configuration was identical to the proposed configuration for the simulation scenarios. In particular, all sensors were generating data every 30 s. In this case, measures about the proposal's precision were obtained as the mean square error (73)–(74) as well. The experiment was also performed for the two proposed operation modes: offline data curation and real-time data curation. The proposed experiment was repeated for different relative lengths of the expanded time interval $[n_1^e, n_2^e]$ with respect to the curation interval $[n_1, n_2]$ as performed in the third experiment. In this case, malfunctions were introduced by actual environmental and technological factors. For each configuration, the system was operating 24 h (so results in the third and fourth experiments are comparable).

For all these experiments, the configuration for the proposed prediction-correction algorithm is shown in Table 2.

Table 2. Proposed algorithm configuration.

Parameter	Value	Comments
E_b	0.2 dBm	Typical value according to the hardware capabilities of ESP-32 microprocessor
γ_{lim}	4	Standard value for the correction capacity of cyclic codes
Λ_{max}	2^{16}	Associated to a 16-bit architecture
Σ	4096	Associated to a 12-bit ADC
Γ	1 h	Standard value in traffic theory
β	3	Typical mathematical order for high-precision applications

4.2. Results

Figure 3 shows the results of the first experiment. As can be seen, for all Industry 4.0 systems up to 10 sensor nodes, the proposed solution was able to curate all data series in real time, as the computational time was below the sampling period. However, systems with higher sample-generation rates ($T_i = 1$ s and $T_i = 5$ s) got congested when the number of nodes went up. Specifically, systems with 20 and 40 elements, respectively, caused the system to become unavailable. Any other higher sampling period did not cause congestion in the system, even with platforms including up to 100 sensor nodes.

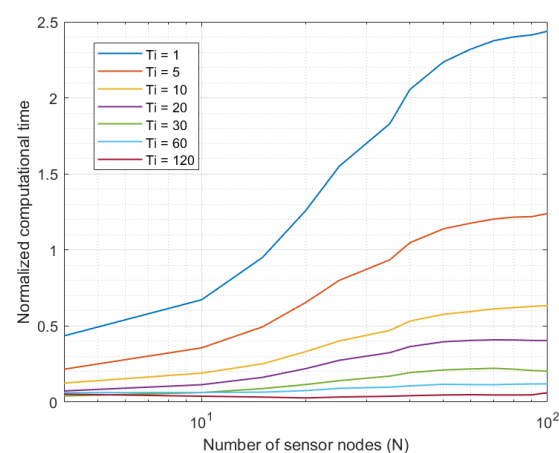


Figure 3. Results of the first experiment.

Although for higher device densities and higher amounts of sensor nodes, the proposed system may get congested with higher sampling periods, 100 nodes is above the number of nodes most current Industry 4.0 platforms include.

Figure 4 shows the results of the second experiment. In this case, we evaluated the computational cost in the offline data curation. As can be seen, in only one case was the computational time above the sampling period: for networks with 100 sensor nodes. This situation was caused by a very populated network (100 sensor nodes), where data were generated at a high rate; greater than the processing rate offered by the prediction-correction mechanism (the data generation rate went up when the number of nodes in the network increased). Thus, after a few samples, the system did not process data “on the fly” and the new data were queued. Therefore, long term, the medium computational time was higher than the sampling period, because of data queues and the congestion caused by the high data-generation rate. In any case, in general and as said before, 101 is a number of devices above the current needs of common Industry 4.0 deployments (although future pervasive platforms may go beyond this number).

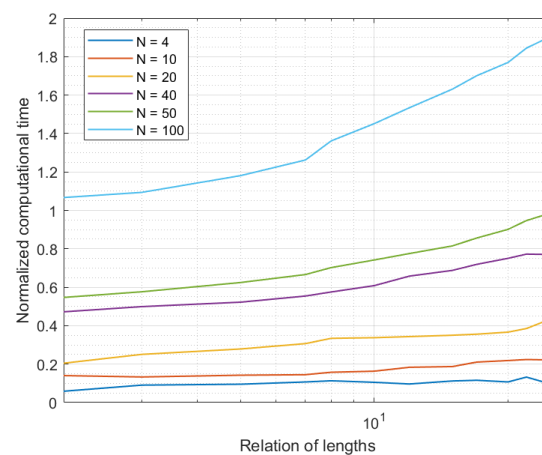


Figure 4. Results of the second experiment.

Only a second system configuration may cause the system to get congested: for an Industry 4.0 system with 50 devices and an extended time interval 25 times higher than the curation interval. Although this is a large number of samples to process, the reduced complexity of the proposed solution allowed (even in this case) computational times slightly below the sampling period.

As a conclusion of these two initial experiments, the proposed solution successfully operated in real time and offline. Besides, it is scalable to large systems, above the current Industry 4.0 needs. However, for future pervasive sensing platforms, the proposed solution may require powerful cloud systems or the definition of a distributed or edge computing scheme.

Figure 5 presents the results of the third experiment. As can be seen, in all situations the MSE was below 50%, even for weak configurations, such as expanded time intervals that were only less than two times higher than the curation interval. As the number of samples that were integrated into the candidate calculation process went up, the MSE clearly went down. For configurations where the length of the expanded interval was around 25 times the length of the curation interval, the MSE was almost null, and the remaining error may be considered residual and intrinsic to all processing schemes.

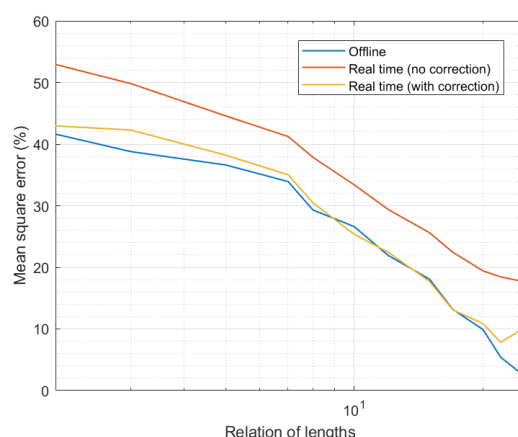


Figure 5. Results of the third experiment.

It was also interesting to analyze the impact of the proposed correction phase. As can be seen, real-time curation showed worse performance than offline data curation, as the available information was more reduced. The MSE may be between 10% and 20% higher in real-time data curation than in offline curation for the same system configuration. However, when considering the proposed correction phase, which was able to integrate future information in the previously obtained curated data series, real-time data curation may reach the same precision as offline mechanisms.

Figure 6 shows a comparison between results obtained in the third experiment and results reported by state-of-the-art solutions [56]. As state-of-the-art mechanisms do not depend on the length of the expanded time interval (this notion was defined only in the new proposed scheme), the MSE for the previously reported solution was constant. Small variations displayed in Figure 6 were caused by numerical effects in the simulation. As can be seen, for expanded time intervals very similar in length to the curation interval, the MSE was lower in state-of-the-art solutions. Decision trees in state-of-the-art mechanisms, to be trained, analyze large amounts of data, so their models are much more precise than the models generated in the proposed framework. However, as larger expanded intervals were considered, the MSE in the proposed solution was greatly reduced, while state-of-the-art mechanisms remained constant. In fact, for relation of lengths above 20, the proposed solution showed an MSE 50% lower than the previously reported mechanism. In conclusion, for models obtained from the analysis of comparable amounts of samples, the proposed framework produced curated data series 50% more precise.

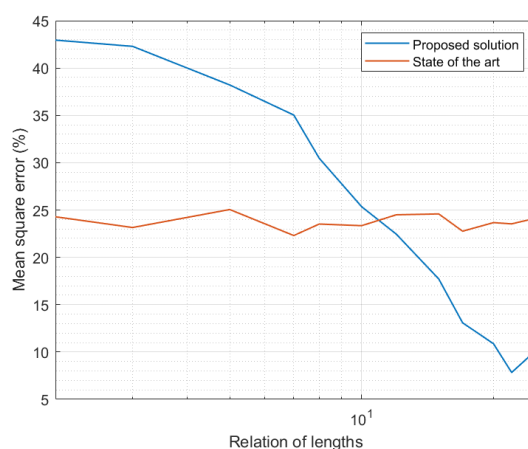


Figure 6. Results of the third experiment: comparison with state-of-the-art solutions [56].

If we evaluate together Figures 3–6, we can conclude the proposed solution successfully obtained a precise curated data series in Industry 4.0 platforms. Besides, if we assume

an error around 5%, the proposed algorithm may be deployed in scenarios with a large number of sensor nodes, above the current state of the art in Industry 4.0 solutions.

Finally, Figure 7 shows the results of the fourth experiment. As can be seen, the evolution of the MSE with the relative interval length was similar to the one observed in Figure 4 (third experiment). As larger intervals were considered and more samples introduced in the candidate calculation process, the precision went up and the MSE went down. However, in this case, the precision was slightly lower than in the simulation study; in particular, we can see the MSE was around 5% higher. In any case, the qualitative evolution of the proposed mechanism was equivalent in simulation and real scenarios. Only in one situation was the performance different in a relevant manner: for curation and expanded time intervals with the same length (relation of lengths equal to the unit). Under those circumstances, in simulation scenarios, the correction phase had a smooth behavior and it still reduced the MSE. However, in real scenarios, when the relation of lengths was equal to the unit, not enough information was provided to the correction algorithm and thresholds did not converge properly. As a consequence, the correction phase introduced an additional error instead of improving the data quality. This situation, in any case, was very transitory and solved immediately when the expanded interval was even slightly higher than the curation interval (relation of lengths above the unit). Despite this fact, and once more, the impact of the correction phase was relevant, even more in this real hardware implementation (MSE may grow up to around 90% without the correction phase).

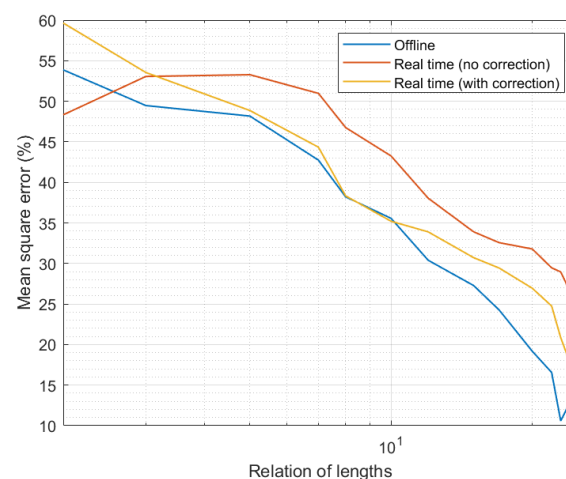


Figure 7. Results of the fourth experiment.

Thus, results allowed us to conclude the proposed solution was successful and valid as a data curation technology for Industry 4.0, focused on improving sensor interoperability.

5. Conclusions

At the software level, real-time algorithms and automatic decision-making mechanisms are very sensitive to the quality of received data. Common malfunctions in sensor nodes, such as delays, numerical errors, corrupted data or inactivity periods, may cause a critical problem if an inadequate decision is made based on those data. The most common solution to this problem is the adaptation and transformation of high-level software components to tolerate these effects, but this calibration turns interoperability between physical sensors and software modules into a very problematic issue.

Therefore, new solutions that guarantee the interoperability of all sensors with the software elements in Industry 4.0 solutions are needed. In this paper, we proposed a solution based on numerical algorithms following a predictor-corrector architecture. Using a combination of techniques, such as Lagrange polynomial and Hermite interpolation, data series may be adapted to the requirements of Industry 4.0 software algorithms. Series may

be expanded, contracted or completed using predicted samples, which are later updated and corrected using the real information (if received).

Through this process, the resulting curated time series has enough quality to be employed with any software module (artificial intelligence, decision making, etc.), guaranteeing the interoperability of all sensor nodes with the high-level applications (which now do not require any adaptation or calibration procedure).

Results show the proposed solution successfully operated in real time and offline and it was scalable to large systems, above the current Industry 4.0 needs. Besides, we can conclude the proposed solution successfully obtained a precise curated data series in Industry 4.0 platforms, even in scenarios with large number of sensor nodes.

Future works will consider the proposed solution in large Industry 4.0 deployments with intense industrial activity, where the environment is more hostile and the operation conditions are more critical.

Author Contributions: Conceptualization, B.B. and T.R.; methodology, B.B.; software, R.A.; validation, B.B., R.A. and T.R.; formal analysis, B.B.; investigation, B.B., R.A. and T.R.; resources, T.R.; data curation, B.B. and R.A.; writing—original draft preparation, B.B.; writing—review and editing, R.A.; visualization, R.A.; supervision, T.R.; project administration, R.A.; funding acquisition, T.R. All authors have read and agreed to the published version of the manuscript.

Funding: The research leading to these results has received funding from the Ministry of Science, Innovation and Universities through the COGNOS project (PID2019-105484RB-I00).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mendelsohn, R.; Neumann, J.E. (Eds.) *The Impact of Climate Change on the United States Economy*; Cambridge University Press: Cambridge, UK, 2004.
2. Chomsky, N.; Pollin, R.; Polychroniou, C.J. *Climate Crisis and the Global Green New Deal: The Political Economy of Saving the Planet*; Verso: London, UK, 2020.
3. Ghobakhloo, M. Industry 4.0, digitization, and opportunities for sustainability. *J. Clean. Prod.* **2020**, *252*, 119869. [\[CrossRef\]](#)
4. Bordel, B.; Alcarria, R. Trust-enhancing technologies: Blockchain mathematics in the context of Industry 4.0. In *Advances in Mathematics for Industry 4.0*; Academic Press: Cambridge, UK, 2021; pp. 1–22.
5. Bordel, B.; Alcarria, R.; Hernández, M.; Robles, T. People-as-a-Service Dilemma: Humanizing Computing Solutions in High-Efficiency Applications. *Multidiscip. Digit. Publ. Inst. Proc.* **2019**, *31*, 39. [\[CrossRef\]](#)
6. Bordel, B.; Alcarria, R. Assessment of human motivation through analysis of physiological and emotional signals in Industry 4.0 scenarios. *J. Ambient. Intell. Humaniz. Comput.* **2017**, 1–21. [\[CrossRef\]](#)
7. Bordel, B.; Alcarria, R.; Robles, T.; Martín, D. Cyber-physical systems: Extending pervasive sensing from control theory to the Internet of Things. *Pervasive Mob. Comput.* **2017**, *40*, 156–184. [\[CrossRef\]](#)
8. Lee, J.; Davari, H.; Singh, J.; Pandhare, V. Industrial Artificial Intelligence for industry 4.0-based manufacturing systems. *Manuf. Lett.* **2018**, *18*, 20–23. [\[CrossRef\]](#)
9. Robles, T.; Bordel, B.; Alcarria, R.; de Andrés, D.M. Mobile Wireless Sensor Networks: Modeling and Analysis of Three-Dimensional Scenarios and Neighbor Discovery in Mobile Data Collection. *Ad Hoc Sens. Wirel. Netw.* **2017**, *35*, 67–104.
10. Bordel, B.; Alcarria, R.; Robles, T. Controlling Supervised Industry 4.0 Processes through Logic Rules and Tensor Deformation Functions. *Informatica* **2021**, *32*, 1–29.
11. Martín, D.; Bordel, B.; Alcarria, R. Automatic detection of erratic sensor observations in Ami Platforms: A statistical approach. *Multidiscip. Digit. Publ. Inst. Proc.* **2019**, *31*, 55. [\[CrossRef\]](#)
12. Martín, D.; Fuentes-Lorenzo, D.; Bordel, B.; Alcarria, R. Towards Outlier Sensor Detection in Ambient Intelligent Platforms—A Low-Complexity Statistical Approach. *Sensors* **2020**, *20*, 4217. [\[CrossRef\]](#)
13. Bordel, B.; Miguel, C.; Alcarria, R.; Robles, T. A hardware-supported algorithm for self-managed and choreographed task execution in sensor networks. *Sensors* **2018**, *18*, 812. [\[CrossRef\]](#)
14. Bordel, B.; Alcarria, R.; Robles, T.; You, I. A predictor-corrector algorithm based on Laurent series for biological signals in the Internet of Medical Things. *IEEE Access* **2020**, *8*, 109360–109371. [\[CrossRef\]](#)
15. Burns, T.; Cosgrove, J.; Doyle, F. A Review of Interoperability Standards for Industry 4.0. *Procedia Manuf.* **2019**, *38*, 646–653. [\[CrossRef\]](#)

16. De Melo, P.F.S.; Godoy, E.P. Controller Interface for Industry 4.0 based on RAMI 4.0 and OPC UA. In Proceedings of the 2019 II Workshop on Metrology for Industry 4.0 and IoT (MetroInd4. 0&IoT), Naples, Italy, 4–6 June 2019; IEEE: Naples, Italy, 2019; pp. 229–234.
17. Kong, X.T.; Yang, X.; Peng, K.L.; Li, C.Z. Cyber physical system-enabled synchronization mechanism for pick-and-sort ecommerce order fulfilment. *Comput. Ind.* **2020**, *118*, 103220. [\[CrossRef\]](#)
18. Geraci, A. *IEEE Standard Computer Dictionary: Compilation of IEEE Standard Computer Glossaries*; IEEE Press: Piscataway, NJ, USA, 1991.
19. Bartodziej, C.J. The concept industry 4.0. In *The Concept Industry 4.0*; Springer Gabler: Wiesbaden, Germany, 2017; pp. 27–50.
20. Adolphs, P.; Bedenbender, H.; Dirzus, D.; Ehlich, M.; Epple, U.; Hankel, M.; Wollschlaeger, M. *Reference Architecture Model Industrie 4.0 (Ram4. 0)*; Status report; ZVEI and VDI: Berlin, Germany, 2015.
21. Pai, D.M. *Interoperability between IIC Architecture & Industry 4.0 Reference Architecture for Industrial Assets*; Tech. Rep.: Piscataway, NJ, USA, 2016.
22. Villalonga, A.; Beruvides, G.; Castaño, F.; Haber, R.E. Cloud-based industrial cyber-physical system for data-driven reasoning: A review and use case on an industry 4.0 pilot line. *IEEE Trans. Ind. Inform.* **2020**, *16*, 5975–5984. [\[CrossRef\]](#)
23. Xia, W.; Zhang, J.; Quek, T.Q.; Jin, S.; Zhu, H. Mobile edge cloud-based industrial internet of things: Improving edge intelligence with hierarchical SDN controllers. *IEEE Veh. Technol. Mag.* **2020**, *15*, 36–45. [\[CrossRef\]](#)
24. Babbar, H.; Rani, S.; Singh, A.; Abd-Elnaby, M.; Choi, B.J. Cloud Based Smart City Services for Industrial Internet of Things in Software-Defined Networking. *Sustainability* **2021**, *13*, 8910. [\[CrossRef\]](#)
25. Li, M.; Xu, G.; Lin, P.; Huang, G.Q. Cloud-based mobile gateway operation system for industrial wearables. *Robot. Comput.-Integr. Manuf.* **2019**, *58*, 43–54. [\[CrossRef\]](#)
26. Tannahill, B.K.; Jamshidi, M. System of Systems and Big Data analytics—Bridging the gap. *Comput. Electr. Eng.* **2014**, *40*, 2–15. [\[CrossRef\]](#)
27. Sun, S.; Zheng, X.; Villalba-Díez, J.; Ordieres-Meré, J. Data handling in industry 4.0: Interoperability based on distributed ledger technology. *Sensors* **2020**, *20*, 3046. [\[CrossRef\]](#)
28. Negri, E.; Fumagalli, L.; Macchi, M. A Review of the Roles of Digital Twin in CPS-Based Production Systems. *Value Based Intell. Asset Manag.* **2020**, 291–307.
29. Qi, Q.; Tao, F.; Hu, T.; Anwer, N.; Liu, A.; Wei, Y.; Nee, A.Y.C. Enabling technologies and tools for digital twin. *J. Manuf. Syst.* **2019**, *58*, 3–21. [\[CrossRef\]](#)
30. Leng, J.; Zhang, H.; Yan, D.; Liu, Q.; Chen, X.; Zhang, D. Digital twin-driven manufacturing cyber-physical system for parallel controlling of smart workshop. *J. Ambient. Intell. Humaniz. Comput.* **2019**, *10*, 1155–1166. [\[CrossRef\]](#)
31. Haag, S.; Anderl, R. Digital twin—Proof of concept. *Manuf. Lett.* **2018**, *15*, 64–66. [\[CrossRef\]](#)
32. Raptis, T.P.; Passarella, A.; Conti, M. Data management in industry 4.0: State of the art and open challenges. *IEEE Access* **2019**, *7*, 97052–97093. [\[CrossRef\]](#)
33. Reis, M.S.; Gins, G. Industrial process monitoring in the big data/industry 4.0 era: From detection, to diagnosis, to prognosis. *Processes* **2017**, *5*, 35. [\[CrossRef\]](#)
34. De Vita, F.; Bruneo, D.; Das, S.K. On the use of a full stack hardware/software infrastructure for sensor data fusion and fault prediction in industry 4.0. *Pattern Recognit. Lett.* **2020**, *138*, 30–37. [\[CrossRef\]](#)
35. Díez-Oliván, A.; Del Ser, J.; Galar, D.; Sierra, B. Data fusion and machine learning for industrial prognosis: Trends and perspectives towards Industry 4.0. *Inf. Fusion* **2019**, *50*, 92–111. [\[CrossRef\]](#)
36. Grangel-González, I. Semantic data integration for industry 4.0 standards. In *European Knowledge Acquisition Workshop*; Springer International Publishing: Cham, Switzerland, November 2016; pp. 230–237.
37. Loures, E.D.F.R.; dos Santos, E.A.P.; Deschamps, F. Data Fusion for Industry 4.0: General Concepts and Applications. In *Proceedings of the 25th International Joint Conference on Industrial Engineering and Operations Management—IJCIEOM: The Next Generation of Production and Service Systems, March 2020. Novi Sad, Serbia, 15–17 July 2019*; Springer Nature: Basingstoke, UK, 2020; p. 362.
38. Kabashkin, I.; Kundler, J. Reliability of sensor nodes in wireless sensor networks of cyber physical systems. *Procedia Comput. Sci.* **2017**, *104*, 380–384. [\[CrossRef\]](#)
39. Hessner, K.G.; El Naggar, S.; von Appen, W.J.; Strass, V.H. On the reliability of surface current measurements by X-Band marine radar. *Remote. Sens.* **2019**, *11*, 1030. [\[CrossRef\]](#)
40. Guna, J.; Jakus, G.; Pogačnik, M.; Tomažič, S.; Sodnik, J. An analysis of the precision and reliability of the leap motion sensor and its suitability for static and dynamic tracking. *Sensors* **2014**, *14*, 3702–3720. [\[CrossRef\]](#) [\[PubMed\]](#)
41. AboElFotouh, H.M.; Iyengar, S.S.; Chakrabarty, K. Computing reliability and message delay for cooperative wireless distributed sensor networks subject to random failures. *IEEE Trans. Reliab.* **2005**, *54*, 145–155. [\[CrossRef\]](#)
42. Castaño, F.; Strzelczak, S.; Villalonga, A.; Haber, R.E.; Kossakowska, J. Sensor reliability in cyber-physical systems using internet-of-things data: A review and case study. *Remote. Sens.* **2019**, *11*, 2252. [\[CrossRef\]](#)
43. Pak, J.M.; Ahn, C.K.; Shmaliy, Y.S.; Lim, M.T. Improving reliability of particle filter-based localization in wireless sensor networks via hybrid particle/FIR filtering. *IEEE Trans. Ind. Inform.* **2015**, *11*, 1089–1098. [\[CrossRef\]](#)

44. Nilsson, J.; Sandin, F. Semantic interoperability in industry 4.0: Survey of recent developments and outlook. In Proceedings of the 2018 IEEE 16th International Conference on Industrial Informatics (INDIN), Porto, Portugal, 18–20 July 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 127–132.
45. Kumar, V.R.S.; Khamis, A.; Fiorini, S.; Carbonera, J.L.; Alarcos, A.O.; Habib, M.; Olszewska, J.I. Ontologies for industry 4.0. *Knowl. Eng. Rev.* **2019**, *34*, e17. [[CrossRef](#)]
46. Kabugo, J.C.; Jämsä-Jounela, S.L.; Schiemann, R.; Binder, C. Industry 4.0 based process data analytics platform: A waste-to-energy plant case study. *Int. J. Electr. Power Energy Syst.* **2020**, *115*, 105508. [[CrossRef](#)]
47. Bordel, B.; Alcarria, R.; Robles, T.; Sánchez-Picot, Á. Stochastic and information theory techniques to reduce large datasets and detect cyberattacks in Ambient Intelligence Environments. *IEEE Access* **2018**, *6*, 34896–34910. [[CrossRef](#)]
48. Kaupp, L.; Beez, U.; Hülsmann, J.; Humm, B.G. Outlier detection in temporal spatial log data using autoencoder for industry 4.0. In *Proceedings of the International Conference on Engineering Applications of Neural Networks, Crete, Greece, May 24–26 2019*; Springer International Publishing: Cham, Switzerland, 2019; pp. 55–65.
49. Kumar, A.; Sharma, D.K. An optimized multilayer outlier detection for internet of things (IoT) network as industry 4.0 automation and data exchange. In *International Conference on Innovative Computing and Communications*; Springer: Singapore, 2001; pp. 571–584.
50. Sik, D.; Levendovszky, J. Detecting outliers and anomalies to prevent failures and accidents in Industry 4.0. In Proceedings of the 2020 11th IEEE International Conference on Cognitive Infocommunications (CogInfoCom), online, 23–25 September 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 000355–000358.
51. Hoppenstedt, B.; Reichert, M.; Kammerer, K.; Spiliopoulou, M.; Pryss, R. Towards a Hierarchical Approach for Outlier Detection in Industrial Production Settings. In Proceedings of the First International Workshop on Data Science for Industry 4.0. Workshop Proceedings of the EDBT/ICDT 2019 Joint Conference, Lisbon, Portugal, 26 March 2019.
52. Polge, J.; Robert, J.; Le Traon, Y. A case driven study of the use of time series classification for flexibility in industry 4.0. *Sensors* **2020**, *20*, 7273. [[CrossRef](#)] [[PubMed](#)]
53. McKinley, S.; Levine, M. Cubic spline interpolation. *Coll. Rediv.* **1998**, *45*, 1049–1060.
54. Underhill, M.J.; Brown, P.J. Estimation of total jitter and jitter probability density function from the signal spectrum. In Proceedings of the 18th European Frequency and Time Forum (EFTF 2004), Guildford, UK, 5–7 April 2004; pp. 502–508. [[CrossRef](#)]
55. Popovski, P.; Stefanović, Č.; Nielsen, J.J.; De Carvalho, E.; Angelichinoski, M.; Trillingsgaard, K.F.; Bana, A.S. Wireless access in ultra-reliable low-latency communication (URLLC). *IEEE Trans. Commun.* **2019**, *67*, 5783–5801. [[CrossRef](#)]
56. Mukhopadhyay, S.; Panigrahi, D.; Dey, S. Data aware, low cost error correction for wireless sensor networks. In Proceedings of the 2004 IEEE Wireless Communications and Networking Conference (IEEE Cat. No. 04TH8733), Atlanta, GA, USA, 21–25 March 2004; IEEE: Piscataway, NJ, USA, 2004; Volume 4, pp. 2492–2497.
57. Yan, J.; Zargarani-Yazd, A. IBIS-AMI modelling of high-speed memory interfaces. In Proceedings of the 2015 IEEE 24th Electrical Performance of Electronic Packaging and Systems (EPEPS), San Jose, CA, USA, 25–28 October 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 73–76.