HAL
open science

Implementing self-* autonomic properties in self-coordinated manufacturing processes for the Industry 4.0 context

Manuel Sanchez, E. Exposito, J. Aguilar

▶ **To cite this version:**

Manuel Sanchez, E. Exposito, J. Aguilar. Implementing self-* autonomic properties in self-coordinated manufacturing processes for the Industry 4.0 context. Computers in Industry, 2020, 121, pp.103247. 10.1016/j.compind.2020.103247 . hal-02956700

HAL Id: hal-02956700

https://cnrs.hal.science/hal-02956700v1

Submitted on 22 Aug 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Copyright

Version of Record: https://www.sciencedirect.com/science/article/pii/S0166361520304814
Manuscript_dd8910ced07d0c8ff3f71a5ddcb9ebb4

Implementing self-* autonomic properties in self-coordinated manufacturing processes for the Industry 4.0 context

M. Sánchez[a,b,c,d]* and E. Exposito[b,d] and J. Aguilar[c,d,e]

[a]*Department of Informatics Engineering, Universidad Nacional Experimental del Táchira, San Cristóbal, Venezuela;* [b]*Université de Pau et des Pays de l'Adour, E2S UPPA, LIUPPA, Anglet, France;* [c]*CEMISID, University of the Andes, Mérida, Venezuela;* [d]*Tepuy R+D Group. Artificial Intelligence Software Development. Mérida, Venezuela;* [e] *Department of Informatics and Systems, Universidad EAFIT, Medellin, Colombia.*

mbsanchez@unet.edu.ve, UNET, Edificio C, Dpto. Ing. en Informática, Av. Paramillo Universidad, San Cristóbal 5001, Táchira, Venezuela *corresponding author

© 2020 published by Elsevier. This manuscript is made available under the CC BY NC user license
https://creativecommons.org/licenses/by-nc/4.0/

Implementing self-* autonomic properties in self-coordinated manufacturing processes for the Industry 4.0 context

Industry 4.0 requires high levels of autonomy in order to guarantee the manufacturing processes to achieve production goals. For this, it is needed high levels of coordination, cooperation, and collaboration, such that the manufacturing process' actors can communicate and interoperate. A previous paper proposed three autonomic cycles of data analytics tasks for self-coordination in manufacturing processes. In this paper, we implement one of these autonomic cycles, allowing self-supervising of the coordination process. This autonomic cycle is designed using the MIDANO's methodology, and implemented and tested using an experimental tool that was developed to replay the production process event logs, in order to detect failures and invoke the autonomic cycle for self-healing when needed.

Keywords: Industry 4.0; self-supervising; autonomic computing; process mining; self-coordination.

1. Introduction

The concept of "Industry 4.0" is projected to bring a variety of benefits to the business, such as product customization, efficiency, productivity, quality, among others (Birkel et al., 2019; Oesterreich & Teuteberg, 2016). Lu (2017) affirms that the principles of industry 4.0 are interoperability, virtualization, decentralization, real-time capability, service orientation, and modularity. In that sense, Pedone & Mezgár (2018) sustains that the interoperability of actors allows increasing the flexibility and adaptability of manufacturing systems. Consequently, (Liao et al., 2017) says that in Industry 4.0, the interoperability principle permits the actors of the manufacturing process to exchange information. Particularly, the actors that take place in manufacturing processes has been defined in (Sanchez et al., 2019a, 2019b) as:

- People: Humans behind a human-machine interface (HMI), a wearable device, or social networks (Flemisch et al. 2012). Generally, in Industry 4.0, people are not

directly immersed in the production line, due to the risk that manufacturing operation can represent for their life (Dencker et al., 2009; Gaham et al., 2015). However, people are expected to collaborate, using an HMI, with other actors, in order to improve the production efficiency, enabling factories to become more agile and more competitive (Dencker et al., 2009; Gaham et al., 2015). Besides, human monitoring will always be essential due to other actors are not able to deal with all possible manufacturing scenarios (Pacaux-Lemoine et al., 2017; Romero et al., 2016).

- Data: Databases, unstructured data, or raw data produced by things, services, or humans.
- Things: It represents anything with connectivity capabilities, like sensors, actuators, smartphones, smart vehicles, computers, robots, among others (Xu et al. 2014).
- Services: It means anything that can be accessed using a service interface, like a Database (DBaaS), Knowledge (KaaS), Software (SaaS), Business Processes (BPaaS), among others. (Lee et al., 2015; Vizcarrondo et al. 2017).

Consequently, Leżański (2017) affirms that the conjunction of hardware devices, information technologies, and the control theory, allows increasing the autonomy of automated mechanical systems. Moreover, in the Industry 4.0 context, the organizations must have the autonomy to schedule tasks for maintenance, failure prediction, reconfiguration, and adapt themselves to new requirements and unexpected changes in the manufacturing processes (Li et al., 2017; Santos et al., 2017).

In past researches, we proposed to use autonomic computing in combination with the information and communication paradigms to deal with the integrability and interoperability challenges in Industry 4.0 (Sanchez et al., 2019b). This combination of

technologies will allow incrementally, adding self-* properties, such as self-connection, self-communication, self-coordination, self-cooperation, and self-collaboration (Sanchez et al., 2019b). Like autonomy and autonomic concepts give the impression of having the same meaning, it is appropriate to clarify these terms. An autonomous system/process refers to a system/process that can be executed from start to finish without human intervention (Collier, 2002; Truszkowski et al., 2010). On the other hand, the autonomic term is derived from autonomous, and it relates to a metaphor-based on biology, specifically, to the ability of the Autonomous Nervous System to reflex reactions involuntarily (Morris, 1982; Sterritt & Hinchey, 2005; Truszkowski et al., 2010). Consequently, Truszkowski et al. (2010) affirm that autonomy means self-governance/self-direction, but autonomic is a specialized form of autonomy for self-management (that means, self-heal, self-protect, self-configure, self-optimize, self-* of the process).

Notably, in previous researches (Sanchez et al., 2019a, 2019b), we have proposed an approach to solve the Industry 4.0 integrability and interoperability challenges incrementally using a stack of five levels, called the 5C stack levels. In this approach, we must start solving the challenges at the level of **C**onnection, next at the **C**ommunication level, and finally, once that actors can adequately connect and communicate, we can solve challenges at the levels of **C**oordination, **C**ooperation, and **C**ollaboration, depending on the actors and system needs. In this way, we can group challenges and deal with them at the proper level. The 5C levels were defined in (Sanchez et al., 2019a, 2019b) as:

- Connection: links the actors to the network, which means that the actors can contact each other (Kumar et al., 2020). The connection is essential to allow communication.

- Communication: lets actors exchange messages, establish a conversation, and interact with other actors (Liu et al., 2014; S.-W. Yang & Chen, 2013). Also, communication means that actors can understand each other. Connection and Communication are essentials to achieve interoperability of the system, as well as to allow more elaborated processes like coordination, cooperation, and collaboration.
- Coordination: is an activity carried out by a central actor (or orchestrator) that allows coherently to harmonizing the execution of the tasks of a system (intra-systems integration or vertical integration) (Pietrewicz, 2019; Suali et al., 2017). In terms of services, coordination is closely related to the concept of intra-system orchestration (internal to a business process or system) (Haupert et al. 2017).
- Cooperation: According to (Berdal et al., 2019; Pacaux-Lemoine et al., 2017), cooperation consists of a negotiation process that allows achieving agreements to the actors of the same system (intra-system integration or vertical integration), or the entities of two or more systems (inter-systems integration or horizontal integration), for the execution of their tasks, in order to accomplish individual objectives. Cooperation is related to inter-system orchestration (Haupert et al. 2017).
- Collaboration: refers to actors of two or more systems (inter-system) that work together in order to achieve a common goal that participants would not be able to accomplish alone (Dencker et al., 2009; Wang et al., 2017). Collaboration is related to inter-system choreography (interactions between autonomous processes) (Haupert et al. 2017). Collaboration does not rely on a central coordinator.

Consequently, the Autonomic computing is a paradigm that allows creating flexible, scalable, and adaptive systems (IBM, 2004; Lalanda et al., 2013; Parashar & Hariri, 2005; Sterritt & Hinchey, 2005; Vizcarrondo et al., 2012). Those systems can change their behaviors, according to their needs, through self-awareness and self-reference, by using introspection (the system's ability to monitor and reason about its internal status (Vizcarrondo et al., 2017)) and intersection (the program's ability to change its execution state (Vizcarrondo et al., 2017))) mechanisms, in order to reason and make decisions. Based on this idea, Sanchez, Exposito, & Aguilar (2019a) proposed a framework for autonomous integration of actors in manufacturing processes, in the context of Industry 4.0. This framework combines the Internet of Everything (IoE) (Martino et al., 2018; L. T. Yang et al., 2017) as an integration layer, and the Autonomic computing and Everything Mining as a reflective layer, in order to promote the autonomic process of self-coordination, self-cooperation and self-collaboration.

The same work proposes three autonomic cycles of data analytics tasks to promote the self-coordination of actors in manufacturing processes. The main idea is that the product being manufactured can coordinate its production, giving instructions to other actors on how coordinately to produce itself. An autonomic cycle of data analytics tasks is defined as a set of data analytics tasks that interoperate together, in order to achieve the objectives that satisfy the needs of the managed resources (Aguilar et al., 2017b, 2017c, 2016). These tasks have different roles in the cycle: Observing the process, analyzing and interpreting what happens in it, and making decisions that allow reaching the objective for which the cycle was designed. In this sense, each autonomic cycle in (Sanchez et al., 2019a) has a different goal regarding the autonomic coordination process for manufacturing, which are: self-configuration of the manufacturing process (create a plan for autonomous coordination of actors), self-supervising of the manufacturing process

(detect system failures) and self-healing of the manufacturing process (auto repair the system).

On the other hand, an autonomous supervisory system in Industry 4.0 context is a system that can perform acquisition of data, context-aware data analysis, and evaluation based on both real-time and historical data (Derboul et al., 2018; Tiboni et al., 2019; Y. Xu et al., 2017). These data analysis tasks produce information that can be used to gain the capabilities of self-awareness and self-maintenance (Lee et al., 2015), which contributes considerably to the resilience, automation, and productivity of manufacturing processes, because it is possible to make predictive decisions about machinery failures and machinery deterioration trends (Dinardo et al., 2018; Lee et al., 2014). Xu et al. (2017) establish the importance of having the right diagnostic approach to guarantee the safe operation of the equipment. Furthermore, Leżański (2017) affirms that the automatic supervision of manufacturing processes belongs to the most advanced features of the autonomy of a machine-based system. Besides, other authors insist that the automated supervision of machine-based systems has become a necessity (Cao et al., 2019), due that a supervisory system can increase their autonomy.

Based on the previous ideas, this paper presents the implementation of a self-supervising autonomic cycle for manufacturing (Sanchez et al., 2019a), in a coordination context. Our self-supervising autonomic cycle is a supervisory system that uses the Autonomic computing paradigm and Everything-mining techniques in order to get useful information oriented to detect and manage system failures. This autonomic cycle of self-supervising has the next features:

- It uses two everything-mining techniques: process mining and big data mining.
- The process-mining gets useful insights from the manufacturing process in a variety of forms. Firstly, it discovers the manufacturing process flows (Petri net

or process graph). This graph is used later to show the process behavior graphically so that the people actor can supervise and control the manufacturing process. However, in this research work, we do not cope with people changing the system behavior, because the primary goal of this research is to provide a self-supervisory system, as a first step to enable autonomic coordination in manufacturing systems. Other useful information provided by the Process mining is the bottlenecks found in the process (actors that probably present issues), and the historical performance of the whole manufacturing process (globally) and actors (individually). This information is crucial in order to detect future failures in the execution of the actor's tasks.

- The data mining builds a predictive model that is used to detect if a product will fail or not the quality control test. This model will allow the system to reconfigure itself in order to avoid or repair the failures.
- The everything-mining paradigm gathers the information that is needed by the self-supervisory system to make decisions.
- The concept of an autonomic cycle using everything-mining techniques has not been used in the past to build a supervisory system for the Industry 4.0 context.

This paper is organized as follows: Section 2 presents the related works, Section 3 details the functional and technical aspects of our proposed architecture, as well as the design of the autonomic cycle for self-supervising. Section 4 shows a case study and the instantiation of the autonomic cycle of self-supervising in it. Section 5, exposes the results, finishing with some conclusions in Section 6.

2. Related Works

This section presents related researches in the field of supervisory systems regarding the Industry 4.0 context. Particularly, Xu et al. (2017) developed a Fault Diagnosis System using the Industrial Big Data concept (Obitko & Jirkovský, 2015). In the first place, they have made a classification of fault diagnosis systems by dividing them into three types.

- Knowledge-driven models: is applyied to system with a small number of inputs and outputs, easy to model, but only for specific type of failures.
- Data-driven models: this model can increase the diagnostic accuracy and the degree of automation using data mining in historical data.
- Value-driven models: similar to data-driven, but this type of fault diagnosis systems use big data and big data analytical methods, to detect particular values that are not easily detected by traditional methods.

In the second place, they introduce a new concept, called Device Electrocardiogram (DEKG), which consists of visualizing every event and motion of the equipment. Moreover, it can monitor the status of operations through the changes in the DEKG, and predict downtimes, in order to provide predictive and proactive maintenance. The architecture of a DEKG fault diagnosis system consists of 4 layers: equipment, data acquisition, processing, and application. The Equipment Layer is the physical layer, where all the equipment is deployed. The Data Acquisition layer gathers and formats massive data from the equipment layer. The data collected in the Data Acquisition layer are transferred to the Processing layer, where various fault diagnosis methods based on big data are used to perform degradation assessment and predictions. Finally, the result of the analysis is sent to the Application layer, where it is used to predict, to generate warning messages, and to optimize the system.

Leżański (2017) develops an architecture of a supervisory system for manufacturing processes in Industry 4.0. Leżański describes the method that a supervisory system must follow for failure detection. The first stage is the sensor stage, in which the data are collected using sensors and send to the next stage. The Signal Processing stage is where the data is processed. The next stage includes feature extraction and selection using Artificial Intelligence methods, find the features that allow the development of the different knowledge models. Those models are used in the Fault detection & the classification stage for failure detection. Finally, in the control strategy stage, a decision-making process takes place, to restore the process to a normal operational state. The architecture proposed by Leżański has five components. The first component groups the two first stages of the supervisory method described previously, and the second component is grouping the last three stages. An adapter is used between the first and second components, to transform the data collected from sensors to the data format used by the second component. Moreover, the output of the second component is transmitted to the client applications, which require specific functionalities of the system.

Silva et al. (2018) create an Intelligent Data Analysis and Real-Time Supervision (IDARTS) framework, with the primary goal of performing data analysis and real-time supervision for manufacturing environments. IDARTS combines distributed data acquisition, machine learning, and run-time reasoning, to assist in fields like predictive maintenance, and quality control. IDARTS was conceived thinking in three principles: the integration of physical and software elements (Thought the application of Cyber-Physical Production Systems (Rojas et al., 2017), abbreviated as CPPS, the data exchange between heterogeneous components (using a common data representation and an exchange format to ensure interoperability), and the knowledge management and data analysis (by employing advanced data analysis and knowledge management methods on

semantically enriched data acquired by the consists Production System). The IDARTS framework is comprised of various modular components. The CPPS is dealing with all the activities related to the acquisition and processing of production data. The CPPS interacts with the Real-time Data Analysis (RDA) component, which is in charge of analysing data during the system's execution, with the purpose of providing relevant information. Finally, the knowledge management component entails the higher-level data analysis and the knowledge generation using the historical data as data source aimed to provide feedback and updates to the previous modules. The IDARTS paper does not show results about the fault detection system.

Tiboni et al. (2019) present a modular architecture for a supervised, fully integrated, and monitored system. The proposed approach is based on commercial devices and an Industrial Internet of Things (Elattar et al., 2017; Haupert et al., 2017; L. Xu et al., 2014) network. The authors do not give a detailed explanation about how to transform a plant into a fully supervised and monitored system, neither about their system architecture. Moreover, they have detailed devices that can be used to enable smart factory technologies.

Furthermore, Derboul et al. (2018) present a study of the impact of a SCADA (Zhou et al., 2017) system on the performance of production processes. The results of this study confirm the positive and direct impact of having industrial supervision in the performance of a production system. The study was conducted in a company in which the usage of the SCADA systems has allowed them to increase their profits, reduce costs, etc. This study is specific to the company that they have studied, which means that this study cannot be generalized to other cases or sectors.

Reis and Gins (2017) provided a study of the evolution of Industrial Process Monitoring (IPM). They conclude that metrics like process-oriented targets (production

throughput, selectivity, product quality) and reliability metrics (service time, down-time, the time between failures, failure rate) have a crucial impact on the global performance of the company, and should not be handled separately. Moreover, they argue that with Prognosis, the IPM will acquire a predictive capability that allows better management of manufacturing processes.

Mangal and Kumar (2016) created a predictive model in the Bosch production line using Big Data Analytics methods to detect what parts are most likely to fail the quality control test. Moreover, they have made an essential analysis of the Bosch Production line datasets, getting useful insights like revealing the anonymized time period, which was encoded by Bosch in a different time unit. Those perceptions are relatively crucial for our research in order to apply the Everything-mining techniques and to build the self-supervisory system correctly.

Cao et al. (2019), has proposed an ontology that formalizes the domain knowledge associated with condition monitoring tasks of manufacturing processes. This ontology can be used in an intelligent condition monitoring system to perform fault prognostics tasks in manufacturing processes. This ontology approach is focused not only on incorporating the knowledge about the prognostic's tasks, but also including the knowledge needed for characterizing the manufacturing actors that are being monitored. The ontology consists of three modules: the Manufacturing module, the Context module, and the Condition Monitoring module. The Manufacturing module represents the domain according to three elements: Product, Process, and Resources. Product and Process are actors involved in the manufacturing process, but the Resources describes the knowledge about the resources used to manufacture a product, and how the Process used them for that purpose. The context module is used to describe the current state of an entity (location, time, activity, and others). The Condition Monitoring module represents the

essential knowledge needed to describe the machinery operation conditions. A reasoning mechanism used over this ontology allows performing machinery state identification and error detection.

On this part of the paper, we are going to use the next set of criteria to compare our approach to the previous works (see Table I):

(1) The number of everything-mining techniques used: This element indicates the number of different mining techniques used to build the supervisory system. It is supposed that each mining technique exploits a different data source to generate a different knowledge, which means that it must be able to detect more different failures.

(2) The type of supervisory system that was built (a. Knowledge-Driven, b. Data-driven, c. Value-Driven), based on the classification proposed by Y. Xu et al. (2017).

(3) They use process mining techniques: process mining can exploit an event log, which is essential for the manufacturing processes in order to get useful insights from the system. Explicitly, the process mining technique can be used off-line to create a precise representation of the real manufacturing process (model). This process model can be fed with new data after each execution of the manufacturing process.

(4) Studied Actors (a. Things, b. Data, c. Services, and d. People): This element indicates the data of the actors that the mining techniques will use. If more actors are studied, then more information and knowledge can be extracted from the system. For the extraction of data from the authors, different mining techniques can be used. For example, a thing minings technique can be considered to analyze the data directly associated with devices (like DEKG). Sentiment analysis and

social networks mining are a kind of people mining due that we can get insights about people's feels, and how that feels can influence their work, cause occupational accidents, among other aspects. Service mining can be used to discover new services. Finally, all mining techniques applied to databases, unstructured data, or raw data not associated directly with a specific actor, are considered as data or semantic mining.

(5) Scalability: Indicates if the supervisory system supports the future inclusion of new mining-techniques or self-* properties. This statement means that the system accepts the inclusion of new models created using other mining techniques, without rewriting all the supervisory system or add it new self-* properties.

(6) The autonomy of the supervisory system: This criterion indicates the level of independence (None, Low, Medium, High, Fully autonomous) of the supervisory system. It means how much the supervisory system can act without human participation.

According to the results shown in Table I, the supervisory systems build in previous researches, still have many issues and need many improvements. For instance, they only use one type of mining technique, mostly data mining techniques. It means that they are not considering useful information that can be found in processes/services, people, and things. Moreover, neither previous research works considered process mining to analyze the process flow in order to improve its autonomy. Regarding the scalability of the system to allow the inclusion of new self-* properties and mining techniques, most of the works are not able to include those capabilities easily; it means that the scalability of those supervisory systems is not good enough in this sense. Concerning the autonomy of the supervisory system, they still need much work oriented to turn-on autonomy in the manufacturing process.

TABLE I. Related works' characteristics

	Criteria					
Research	**1**	**2**	**3**	**4**	**5**	**6**
(Leżański, 2017)	1	c	No	b	No	Low
(Y. Xu et al., 2017)	1	c	No	a	No	Medium
(Tiboni et al., 2019)	0	n/a	No	n/a	No	Low
(Derboul et al., 2018)	0	a	No	b	No	Low
(Cao et al., 2019)	1	b	No	a, c	Yes	Medium
(Silva et al., 2018)	1	c	No	b	Yes	High
(Reis & Gins, 2017)	0	n/a	No	n/a	n/a	None
(Mangal & Kumar, 2016)	1	c	No	b	No	None
Our approach	several	b	Yes	a, b, c, d	Yes	High

In that sense, this work proposes an autonomic cycle for self-supervising, oriented to enable self-coordination in manufacturing processes. It is one of the three autonomic cycles that provide the self-coordination capability to the system. The other two cycles allow self-planning and self-healing features (Sanchez et al., 2019a). Therefore, the previous statement demonstrates the scalability of this system in terms of the inclusion of new autonomous capabilities. Also, our autonomic cycle of self-supervising uses two everything mining techniques (process mining and data mining), which generate different knowledge about the system.

Additionally, the related works remark deficiencies and future works regarding the autonomous supervisory systems for Industry 4.0. For instance:

- Tiboni et al. (2019) say that the description of the supervisory system's implementation using artificial intelligence tools in a whole plant is a missing work. Our paper describes how to combine different artificial intelligence techniques and paradigms, in order to implement a self-supervisory system in a production plant.
- Xu et al. (2017) indicate that it would be valuable to fuse different diagnostic methods to make maintenance decisions automatically. Our proposed self-supervising autonomic cycle combines two different everything-mining

techniques for failure detection and prediction. Besides, one of the everything-mining techniques is a log-based process mining, which, according to the literature review, is a technique that has not been studied to build supervisory systems in the Industry 4.0 context. Moreover, the self-supervisory system implemented in this paper can autonomously make decisions and start the self-repairing mechanisms when a failure is detected or predicted.

- Derboul (2018) suggests that the generalization of the results to other cases is still missing work. In that sense, the present paper describes, not only how our architecture extends the existing reference architectures for Industry 4.0, but also, it shows models, diagrams, and algorithms that could be used to generalize and reproduce the present research.

3. Our Architecture

3.1. Proposed Autonomic Integration framework (AIFI 4.0)

In previous research (Sanchez et al., 2019a), we have proposed a framework based on the Autonomic Computing paradigm (Lalanda et al., 2013; Parashar & Hariri, 2005; Vizcarrondo et al., 2012), the Internet of Everything (IoE), and the Everything-mining (X mining) as crucial elements, to guarantee the autonomy, integrability, and interoperability of the actors involved in manufacturing processes by enabling self-* properties in the system regarding the Industry 4.0 context (Burns et al., 2019; Liao et al., 2017).

Fig. 1 shows the architecture of the proposed framework, which is called AIFI 4.0. AIFI is composed of three layers. The Physical layer corresponds to the manufacturing process itself, where all the actors are involved (Sanchez et al., 2019a).

From the previous paragraph, it can be seen that the actors of the manufacturing process are directly related to the actors of the IoE paradigm. Because of that, the *Integration Layer* is centered on IoE as integration media. IoE guarantees the connectivity of the actors. Besides, the main point of this layer is the integrability of actors in the manufacturing process by allowing them to connect and to communicate. This business process is deployed as a service (BPaaS) in the integration platform. The Integration Layer enhances the technical interoperability by defining the infrastructure and protocols necessary for the communication of the actors. Some previous works in this domain are (Burns et al., 2019; Liao et al., 2017; Liu et al., 2014; Sanchez et al., 2018a, 2018b).

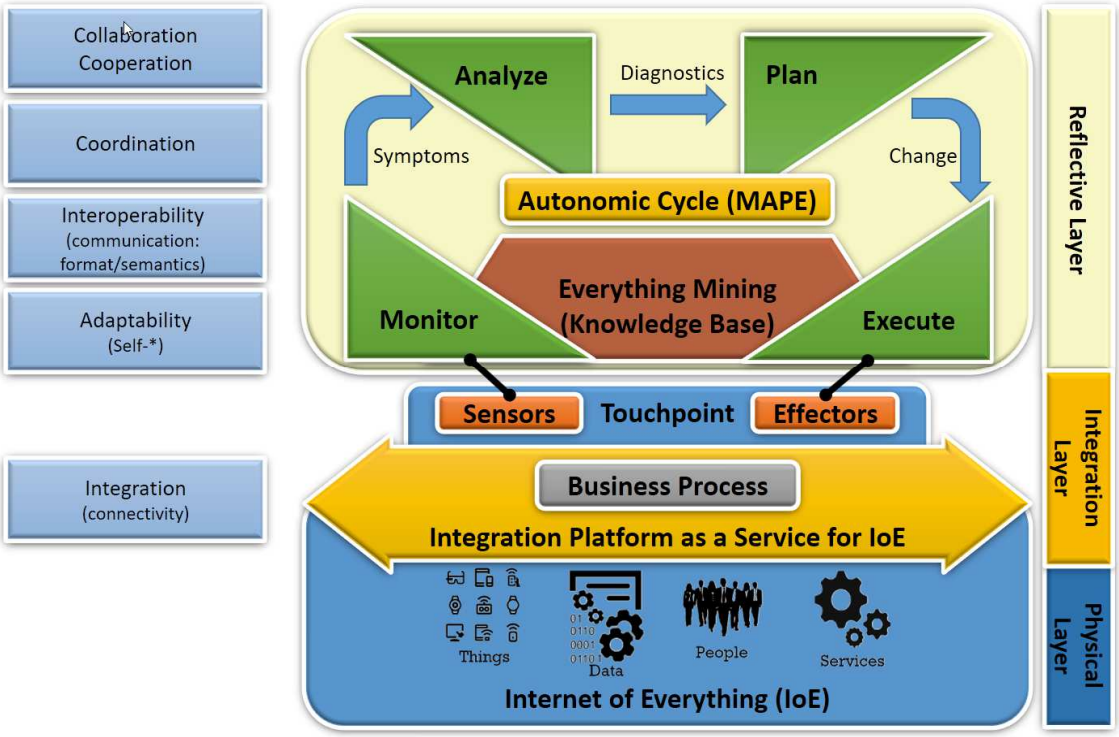

Figure 1. Autonomic Integration framework for Industry 4.0

The *Reflective Layer* uses the Autonomic Computing paradigm, with the primary goal of enhancing the interoperability of actors, so that they can exchange information and use it for self-organization, by enabling self-coordination, self-cooperation, and self-collaboration processes, as autonomic cycles of data analytics tasks. This layer is

responsible for preparing all the knowledge bases needed to allow the functional and semantic interoperability in the system. This interoperability problem has been studied in previous researches (Aguilar et al., 2017d; Burns et al., 2019; Kalatzis et al., 2019; Liao et al., 2017; Obitko & Jirkovský, 2015; Pedone & Mezgár, 2018; Sanchez et al., 2019a; Vizcarrondo et al., 2017). Moreover, this layer allows the deployment of any self-* properties into the system to gain adaptability and autonomy capabilities to the manufacturing process. The autonomic features described previously are designed as autonomic cycles of data analytics tasks. In our case, the Managed Resource is the Business Process because it is the element that needs improving its autonomy. Consequently, each autonomic cycle requires the utilization of Everything-mining techniques, applied to the data sources linked to the production process. The main X-mining techniques are data mining (big data analysis, unstructured data mining, etc.), things mining (Devices mining, etc.), people mining (social network analysis, sentiment analysis, among others), and service mining (process mining, service mining, etc.). These mining techniques create the knowledge base needed by the autonomic cycles to understand the system and to make decisions that might autonomously impact the entire process. Consequently, the X-mining techniques discover useful information (from the actors) for the self-coordination, self-cooperation, and self-collaboration processes, to get an understanding (semantic) about the process to establish a plan defining how/when/where the actors must interact oriented to self-organize themselves and to guarantee an efficient achievement of their individual and collective goals (Sanchez et al., 2019a, 2019b).

In that sense, an autonomic manager that allows self-coordination in manufacturing processes regarding the Industry 4.0 context was proposed by Sanchez et al. (2019a). This autonomic manager comprises three autonomic cycles of data analytics

tasks; they were named ACCI40-*. The first autonomic cycle (ACCI40-1) can build a coordination plan for the production process (self-configuration), based on the production goals and the current context (availability of the entities, their characteristics, etc.). The outcome of this autonomic cycle is the prescriptive model of the coordination plan.

The second autonomic cycle (ACCI40-2) is in charge of the supervision of the execution of the previous plan, to detect failures (self-supervising), and ensure that the plan is being executed correctly. The outcome of this cycle is a system's diagnostic model. The last autonomic cycle (ACCI40-3) is responsible for the reconfiguration of the coordination plan (self-healing) when the ACCI40-2 detects an abnormal situation. ACCI40-3 generates a prescriptive model for the reconfiguration of the current coordinated process.

These three autonomic cycles allow the next self-* properties: self-configuration, self-supervising, and self-healing, for the coordination of actors in the manufacturing process, in order to properly reach the production goals. However, this paper is focused on describing how to use the everything-mining and autonomic computing to enable the self-supervising autonomic cycle in a manufacturing process in the Industry 4.0. This work corresponds to the third layer (coordination) of the 5C integration stack.

3.2. Design of the Autonomic cycle of self-supervising

In this sub-section, we detail the design of the ACCI40-2 (The self-supervising autonomic cycle). This autonomic cycle was designed following MIDANO's methodology (Aguilar et al., 2017a; Pacheco et al., 2014; Rangel et al., 2013). MIDANO is a methodology used for developing data analytics tasks and consists of three phases (see Fig. 2).

Phase 1: The main goal of this phase is knowing the organization, its processes, the experts, among other aspects, such that the goals of the data analytics tasks in the

organization can be set. Moreover, in this phase, the specification of the autonomic cycles for data analysis will be made.

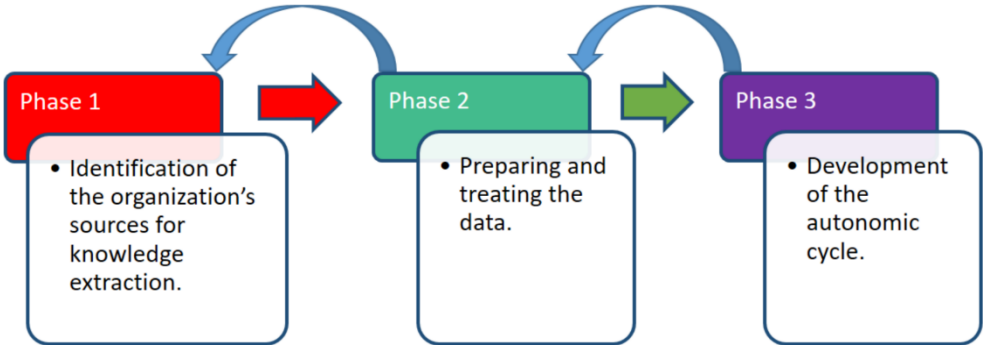

Figure 2. MIDANO's methodology (Aguilar et al., 2017a; Pacheco et al., 2014; Rangel et al., 2013).

Phase 2: This phase is based on an ETL process (Extraction, Transformation, and Load), whose purpose is extracting, transforming and loading the data that will be used by the data analytics tasks. A Minable View (MV) is created for this purpose, which contains all the useful variables to achieve the goals of the autonomic cycles.

Phase 3: In this phase, all the data analytics tasks of the autonomic cycle are implemented. These tasks allow creating the required knowledge models, such as predictive models, descriptive models, etc. This phase ends with the implementation of a prototype of the autonomic cycle.

Mainly, in this paper, we are centered on the autonomic cycle for self-supervising (ACCI40-2). This autonomic cycle consists of three data analytics tasks, which are detailed below:

3.2.1. Task 1: Build/update a model of the production process based on historical data.

The characteristics of this task are listed in Table II. This task uses a Process Mining technique to create a model that allows identifying the desired patterns for fault detection. Essentially, this task allows discovering useful information like the production's flow, problematic stations (bottlenecks), the historical processing time on average for each

station, as well as the global performance of the whole manufacturing process (throughput time). The model created by this task contains all this information, and it is used by Task 2 and 3 for failure detection.

TABLE II. Data Analytics Task 1 Characteristics

Task Name	Build/update a model of the production process based on historical data
Task Description	Create/update the manufacturing process model, which give us useful information that can be used to detect failures
Data source	Historical data gathered by sensors.
Data analytics task type	Association
Data analytics technique	Process Mining
Type of knowledge model	Descriptive model
Related Data analytics tasks	Task 3
Autonomic cycle task type	Analysis

3.2.2. Task 2: Build a predictive model for the quality control test based on historical data.

This data analytics task is focused on failures that are detected using a predictive model based on data about the quality control test results obtained from each product. The predictive model is created using a machine learning technique applied to the quality control test results of the products. This model detects failures before each product enters in the manufacturing line, and allows repairing the system in order to avoid that failure. The characteristics of this task are shown in Table III.

3.2.3. Task 3: Determine how the coordination plan is currently executing.

This task uses the current manufacturing events as input in order to detect failures in the global performance of the manufacturing process. The failure detection is made based on the models created in previous tasks. Firstly, the process model created in Task 1 will help in detecting stations' failures (determine actors that do not guarantee the manufacturing process), as well as failures in the global performance of the production

process. Secondly, the model created in Task 2 is essential to detect whether or not a product will pass the quality control test before starting its production. When this task detects an anomaly, the autonomic cycle of self-healing is invoked, in order to repair the system. The characteristics of this task are shown in Table IV.

TABLE III. Data Analytics Task 2 Characteristics

Task Name	Build a predictive model based on historical data
Task Description	Create a predictive model for failure detection using the quality control test result.
Data source	Historical data containing the quality control test result.
Data analytics task type	Classification
Data analytics technique	Neural networks
Type of knowledge model	Classification model
Related Data analytics tasks	Task 3
Autonomic cycle task type	Analysis

TABLE IV. Data Analytics Task 3 Characteristics

Task Name	Determine which actors do not guarantee the manufacturing process.
Task Description	Detect failures using models created in other data analytics tasks.
Data source	Manufacturing events gathered in the current execution of the manufacturing process.
Data analytics task type	Classification
Data analytics technique	Process mining and data mining
Type of knowledge model	Classification model
Related Data analytics tasks	Task 1 and Task 2
Autonomic cycle task type	Decision-making

Fig. 4 shows the component diagram of the autonomic cycle for the self-supervising prototype. The Business Process is the component that is supervised. The Predictive model is the output of Task 2, while the Process model is the output of Task 1. Finally, the diagnostic module characterizes Task 3. The Business process provides the categorical, numeric, and date features required by the Predictive model, in order to make the quality control test prediction. Similarly, the event log required by the Process model

to detect failures is provided by the Business Process. The diagnostic module uses the event log, the process graph (provided by the Process model), the result from the Predictive model, and the result from the Process model, in order to determine the status of the manufacturing process (decision-making), and invoke the autonomic cycle for self-healing when needed.

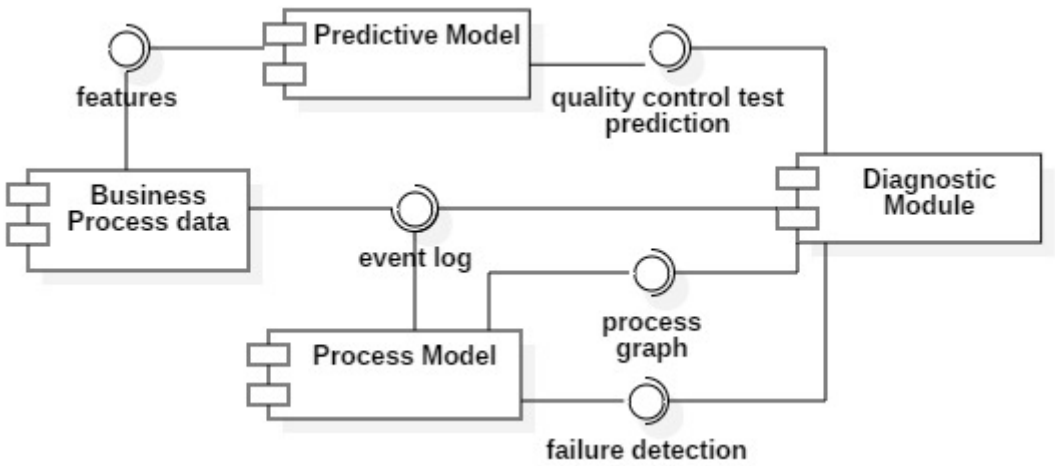

Figure 4. Autonomic cycle of self-configuring (component diagram).

4. Case Study

4.1. Description of the Bosch Production line dataset.

The dataset used in this research for experimentation corresponds to a manufacturing process of auto parts in the Bosch Industry (Kaggle, 2016). Bosch is an enterprise that manufactures parts for car-engines, and it is mainly focused on spark plugs. The manufacturing process is driven by a production line with different stations that are in charge of assembly, test, etc., each product.

According to (Mangal & Kumar, 2016; Singla & Agrawal, 2016), the Bosch production line training dataset contains 1,183,747 samples (it means, auto parts produced). Moreover, the dataset comprises three types of features: 968 numerical features, 2140 categorical features, 1156 date-stamps, and a label indicating if the part is good or bad (the quality control result). However, this data is completely anonymized,

that means that we do not have information about the type of product that is being manufactured, or the goal of each station in the production line, neither if the station corresponds to a device, a person or a service. Nevertheless, (Mangal & Kumar, 2016; Singla & Agrawal, 2016) say that there exist 51 stations distributed among four production lines.

Furthermore, like the data is anonymized, the features are labeled following a convention that tells the production line, the station on the line, and a feature number. E.g., L0_S2_F35 is a feature measured on line 0, station 2, and feature number 35 (Mangal & Kumar, 2016; Singla & Agrawal, 2016). Besides, each date column ends in a number that corresponds to the feature. E.g., the value L3_S51_D4259 is the time at which L3_S51_F4259 was taken (Mangal & Kumar, 2016; Singla & Agrawal, 2016). Finally, it is essential to remark that the data is highly unbalanced; it means that there are 6,879 positive cases (failures) and 1,176,868 negative cases (success).

The Bosch production line also includes a test dataset with 1,183,748 samples. Moreover, this dataset does not contain information about the result of the quality control process. Additionally, the test dataset follows the same conventions as the training dataset.

One crucial point to consider regarding the Bosch Production Line dataset is that it is completely anonymized, which means that the type of contextual information included in the dataset and the type of actors that take part in the manufacturing process are unknown. In that sense, the Everything-mining paradigm is essential to the objective of determining what mining technique will fit the best to extract the information needed by the autonomic cycles, and allow the self-supervising of the manufacturing process.

4.2. Implementation of the autonomic cycle of self-supervising

In this sub-section, it is detailed the implementation of the autonomic cycle described in Section III. The macro algorithm followed to implement this autonomic cycle, as well as the technological tool used, are shown in Table V.

TABLE V. Macro-Algorithms to Implement the Self-Supervising Autonomic Cycle

Data Analytics Task	**Macro-algorithm**	**Tools**
Task 1	1. Extract the manufacturing process event logs from the Bosch training dataset. 2. Apply the process mining algorithm 3. Create a manufacturing process model.	Python pandas, CSV, DateTime, and Numpy, Celonis (Celonis SE, 2019).
Task 2	1. Make the appropriate transformations to the Bosch training dataset. 2. Analyze the training dataset. 3. Select the machine learning technique that best fits the data. 4. Train the predictive model using the Bosch dataset. 5. Verify the model.	Python pandas, Numpy and Keras.
Task 3	1. Extract the event logs from the test dataset. 2. Run each test event log in the process model and look for failures in the performance of the production process. 3. Run each test event log in the process model and look for failures in the stations of the manufacturing process. 4. Run each test event log in the predictive model, and make predictions about the quality control test of each product. 5. If a failure is detected, invoke the autonomic process for self-healing.	Python (Python TM, 2019) pandas and Numpy, C++.

To implement *Task 1* was necessary to transform the Bosch dataset into an event log due that the Process mining algorithm requires the data in the format shown in Fig. 5.

Case ID	Activity	Timestamp	Resource
Id	Task being developed	YYYY-MM-dd HH-mm-ss	Resource involved in the activity

Figure 5. Event logs format required for process mining.

Where:

- Case ID: This is an identifier for the auto-part that is being produced.

- Activity: Represent the stations that constitute the production lines, in which the auto-parts are processed.
- Timestamp: It is a date that indicates when a station starts the processing of the auto-part (see Fig. 3). In the Bosch dataset, the format of that value is a decimal number, and it does not correspond to the required format of date, as shown in Fig. 5.
- Resource: This is an optional parameter that represents a feature, or a resource used in the station that is currently processing the auto-part. This value is not considered in our case.

Consequently, the Bosch manufacturing dataset was transformed to fit the format in Fig 5 (the Python algorithm used, can be found in (Sanchez et al., 2019c)). This format lets us create a minable view that can be used as data-source for Task 1. The resulting minable view is shown in Fig. 6. The significant central transformation made during this step concerns the date on which each product was processed for each station. Bosch anonymized this value as a decimal number in an unknown format. However, Mangal and Kumar (2016) have deduced, after analyzing the correlation of the data, that 0.01 units of this value are equivalent to 6 mins.

Id	Activity	Date
4	L0_S0	2017-02-04 06:24:00
4	L0_S1	2017-02-04 06:24:00
4	L0_S2	2017-02-04 06:24:00
4	L0_S4	2017-02-04 06:36:00
4	L0_S7	2017-02-04 06:36:00

Figure 6. Event logs produced after transforming the Bosch dataset.

Once the data is in the correct format, we proceed to apply the process mining technique using the Celonis tool (Celonis SE, 2019). This step comprises uploading the

training event logs to Celonis, set the parameters, and start the process mining algorithm. Celonis processes the data and returns the discovered process graph.

Using the knowledge discovered by Celonis, we have built a model of the Bosch manufacturing process, that can be used as a knowledge base for decision-making, in order to detect failures in future production events (events from the test dataset) of the manufacturing process. Firstly, we proceeded to extract the information from Celonis and saved it into three CSV files that contain the connection between stations and the stations' processing time. Secondly, it was created a Python script that takes these three files as input and generates a single file with all the process model information (this code is available at (Sanchez et al., 2019c)). The information saved to this model is shown in Fig. 7.

```
process:BOSH Production Line using Trimmed Mean
actors:+BEGIN,L0_S0,L0_S1,L0_S2,L0_S3,L0_S4,L0_S5,L0_S6,L0_S7,
        L0_S8,L0_S9,L0_S10,L0_S11,L0_S12,L0_S13,L0_S14,L0_S15,
        L0_S16,L0_S17,L0_S18,L0_S19,L0_S20,L0_S21,L0_S22,L0_S23,
        L1_S24,L1_S25,L2_S26,L2_S27,L2_S28,L3_S29,L3_S30,
        L3_S31,L3_S32,L3_S33,L3_S34,L3_S35,L3_S36,L3_S37,L3_S38,
        L3_S39,L3_S40,L3_S41,L3_S42,L3_S43,L3_S44,L3_S45,
        L3_S46,L3_S47,L3_S48,L3_S49,L3_S50,L3_S51,-END
process_throughput_time:6420
total_cases:1183165
transitions:
BEGIN=L0_S0;0.0;673862.0;1!L0_S1;0.0;841.0;0!L0_S2;0.0;4.0;0!
      L0_S3;0.0;1.0;0!L0_S4;0.0;5.0;0!L0_S5;0.0;3.0;0!
      L0_S12;0.0;241261.0;1!L0_S13;0.0;4.0;0!L0_S14;0.0;2.0;0!
      L0_S15;0.0;4.0;0!L0_S16;0.0;11.0;0!L0_S17;0.0;11.0;0!
      L0_S18;0.0;8.0;0!L0_S19;0.0;4.0;0!L0_S20;0.0;3.0;0!
      L0_S21;0.0;2.0;0!L0_S22;0.0;1.0;0!L0_S23;0.0;2.0;0!
      L1_S24;0.0;180818.0;1!L1_S25;0.0;82993.0;1!
      L2_S26;0.0;71.0;0!L2_S27;0.0;575.0;0!L2_S28;0.0;150.0;0!
      L3_S29;0.0;2082.0;0!L3_S30;0.0;7.0;0!L3_S35;0.0;5.0;0!
      L3_S36;0.0;10.0;0!L3_S37;0.0;2.0;0!L3_S38;0.0;105.0;0!
      L3_S39;0.0;318.0;0
L0_S0=L0_S1;0.0;673063.0;1!L0_S2;2.0;425.0;0!L0_S3;3.0;372.0;0!
      L0_S4;12.0;2.0;0
L0_S1=L0_S2;2.0;339345.0;1!L0_S3;2.0;334054.0;1!
      L0_S4;18.0;234.0;0!L0_S5;20.0;266.0;0!
      L3_S29;2377.0;5.0;0
```

Figure 7. Bosch production line process model.

As can be seen, it contains the manufacturing process name, the actors (for example, L0_S0 represents the stations number 0 of the production line number 0), the avg. production time, the total number of cases uses to build the model, the connections among actors (for example, L0_S1=L0_S2;2.0;339345 represents a transition from station number 1 to station number 2 in the production line 0), as well as the processing

time of each station (for example, in the transition L0_S1=L0_S2;2.0;339345; the value 2.0 represents the processing time in average of the products, and the value 339345 indicates the number of products used to get that value).

At this point, we can build a process model using a process mining technique. An essential characteristic of our framework is that it allows the model to be validated by an expert, which can also incorporate adjustments or make suggestions oriented to improve the information contained in the models.

The second model (*Task 2*), is a predictive model that was built to detect if an auto-part will fail the quality control test. The predictive model is the second model that Task 3 uses for the decision-making process. This model can be built because the training data contains a column that indicates whether an auto-part fails or pass the quality control test. This classification model can predict both results, positive and false quality control test results, with a high level of confidence. In this way, a Python script was used to train the predictive model using different algorithms. Primarily, we used five different classifiers and selected the one that gave us the best results. Those classifiers are:

- Linear Discriminant Analysis (LDA). It is a commonly used technique for data classification, which is typically employed for dimensionality reduction and pattern identification (Tharwat et al., 2017).
- Balanced Random Forest (BRF). Classical Random Forest is an ensemble of decision trees used for data classification (Pal, 2005). This paper uses a variation of the classical Random Forest Algorithm, which can deal with imbalanced classes (Lemaitre et al., 2014b; O'Brien & Ishwaran, 2019).
- Balanced Bagging (B-B). Bagging consists of an ensemble of classifiers that build several estimators on different subsets of data randomly selected (Breiman, 1996; Buitinck et al., 2013; Pedregosa et al., 2011). A Balanced Bagging classifier adds

an extra step oriented to adequately balance the training dataset (Lemaitre et al., 2014a).

The training dataset comprises 400,000 entries of a total of 1,183,748. Another 100,000 entries are used as the testing set. To validate the classification models, we have used some metrics commonly used for this kind of machine learning models, such as precision, recall, f1-score, accuracy, Matthew's correlation coefficient (MCC), and the Receiver operating characteristic (ROC) curves (Singh et al., 2009; Wardhani et al., 2019). Table VI shows a summary of those metrics.

TABLE VI. Result of metrics used to evaluate the classifiers.

Classifier	Class	Precision	Recall	Accuracy	F1-Score	MCC
B-RF	Pass	100.00%	99.99%	99.99%	100.00%	0.9941
	Fail	98.83%	100.00%	100.00%	99.41%	
B-B	Pass	100.00%	99.99%	99.99%	100.00%	0.9957
	Fail	99.16%	100.00%	100.00%	99.58%	
LDA	Pass	99.31%	99.54%	99.54%	99.42%	0.17977
	Fail	21.92%	15.69%	15.69%	18.29%	

From the previous table, it can be noticed that the Balanced Random Forest algorithm shows good results predicting whether each auto-part passes or fails the quality control test. Besides, the B-RF MCC is 0.9941, which means that this classifier is good predicting both classes (pass and fail). The Balanced Bagging classifier presents similar metric values to the B-RF classifier, which indicates that both are good at predicting positive and negative values. However, the B-B MCC is a little higher than the B-RF MCC, which could indicate that the B-B classifier is better than B-RF.

Finally, the LDA classifier predicts if an auto-parts pass the quality control test with an f1-score of 99.42%, but the f1-score for auto-parts that fails the quality control test is 18.29%. Consequently, the MCC for this classifier is 0.17977, which confirms that this classifier is not good at predicting both classes.

Basically, from the previous discussion, it is clear that the B-RF and B-B classifiers present the best classification rate for both classes (pass and fail). To decide which of these two classifiers will be used by the autonomic cycle, it is needed to use another technique to measure the predictive performance of these classifiers. In this sense, Fig. 8 presents the ROC curves for both classifiers.

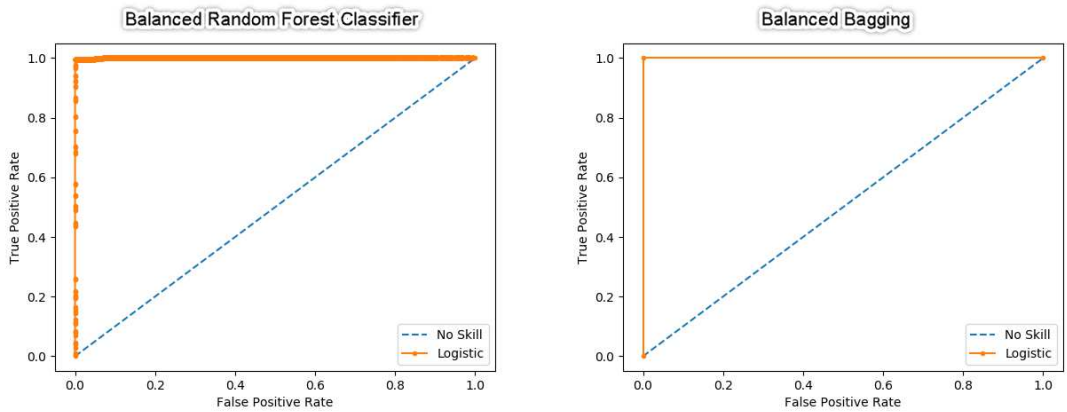

Figure 8. Comparison of B-RF and B-B classifiers using the ROC curves.

The B-RF classifier ROC curve shows that this classifier does not always separate the classes correctly with an Area Under the Curve (AUC) of 0.9969. However, regarding the B-B classifier, the ROC curve shows that it does a perfect class separation with an AUC of 1.0. The previous statement confirms that the B-B classifiers present the best results for this dataset.

In general, if this model gets a false positive (an auto-part that fails the quality control test, but the predictor says it passes), the self-healing autonomic cycle does not be activated, which means that the self-supervisory system does not comply its design goal. If the predictor gets a false negative (an auto-part that passes the quality control test, but the predictor says it will fail), then the production process is reconfigured incorrectly. However, this issue can be rechecked by the self-healing autonomic cycle, in which case

the operator must be involved in order to make the correct decision. The lowest those situations happen, the best the supervisory system works.

Task 3 uses all the data analytics models created in Tasks 1 and 2, in order to detect and predict failures. The prototype of the autonomic cycle for self-supervising was created as a Qt application (The-Qt-Company, 2019). Fig. 9 describes how the people actor can interact with this prototype.

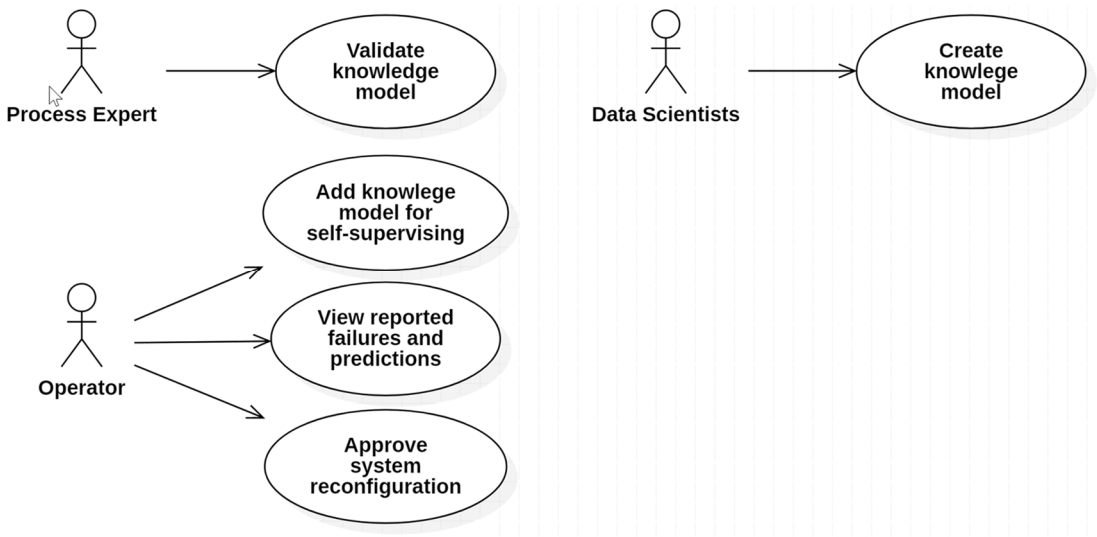

Figure 9. Use case of the people actor.

The Data Scientist is in charge of creating the everything-mining models by applying different mining techniques. For this particular research, the Data scientist generates a process model and a predictive model applying process mining and data mining techniques. The manufacturing process expert must validate the models created by the data scientist to guarantee that they represent the real process. Once the everything-mining models are validated, the operator can use them in the autonomic cycle. One time the autonomic cycle is personalized to the supervised process, the operator can see all the details about the events of failures and quality control test predictions detected.

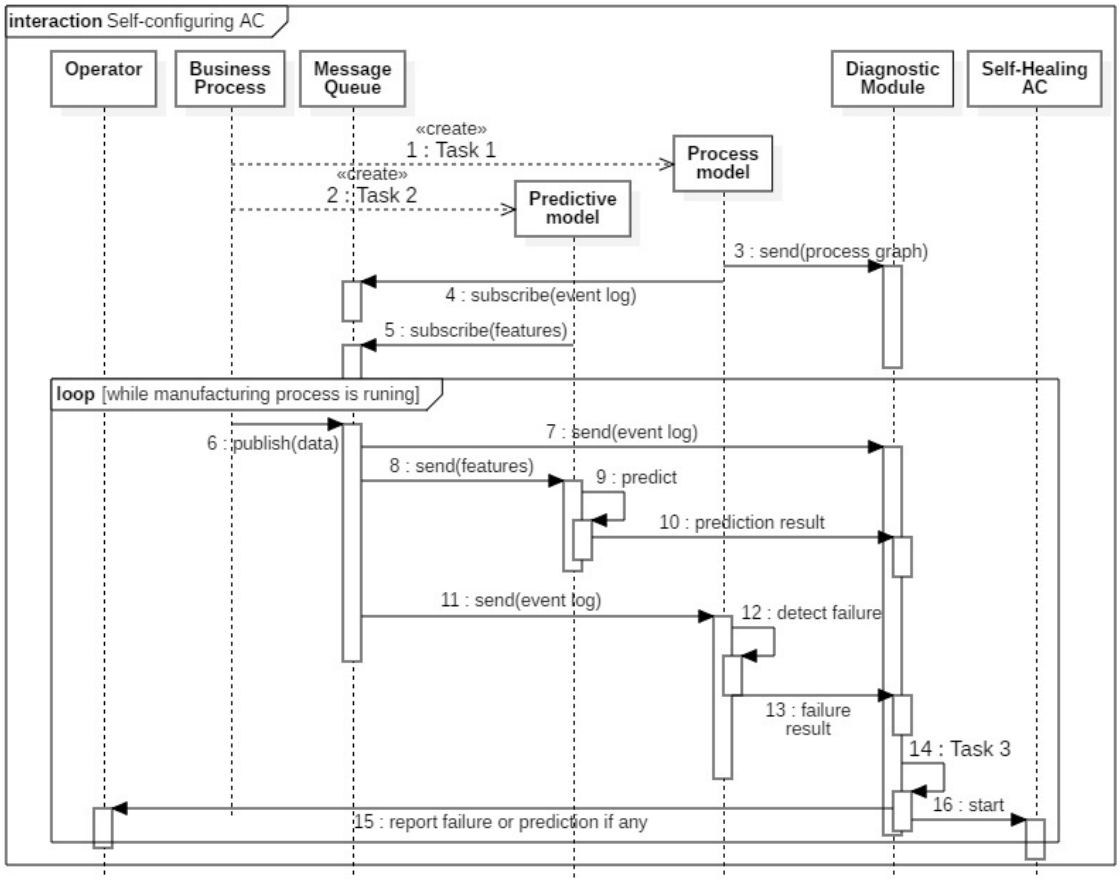

Figure 10. Autonomic cycle of self-supervising (sequence diagram).

Notably, this application uses the predictive model in order to predict quality control test failures and detects station performance failures and global performance failures. Thus, it supports two Everything-mining techniques, process mining and big data mining. Fig. 10 shows, more specifically, the functionality of the autonomic cycle of self-supervising.

Previous to the start of the manufacturing process, Task 1 (message 1) and Task 2 (message 2) are invoked, in order to create or update the process and predictive models, using the historical data of the Business process. When the manufacturing process starts, the Process model component subscribes to the Message queue to get informed about each event that happens during the manufacturing process (message 4). In the same way, the Predictive model subscribes to the Message queue to get informed about the data

required to make quality control test predictions (message 4). At the same time, the Process model sends the process graph to the diagnostic module (message 3), so that it can be aware of the current production process layout.

While the manufacturing process is running, the Business process sends data to the Message queue regularly (message 6). The Message queue module transforms and sends the data to the corresponding data analytics model. The process model receives the data as event logs (message 11), but the predictive model receives it as features (message 8). The process and predictive models process the corresponding data and emit the result to the Diagnostic module (messages 9, 10, 12, and 13). The Diagnostics module sends the results to Task 3 (message 14), in order to determine how the coordination plan is currently executing. If any failure is detected, then it is informed to the operator so that he/she can be aware of the system events and maybe, can make future decisions if that is required (message 15). Next, the autonomic cycle of self-healing is invoked (message 16) in order to repair the manufacturing process. Consequently, the Message Queue element is a component that resides in the communication layer supported by IoE, which allows the integration of the Business Process with the autonomic cycles.

As can be seen, the operator is not involved in the decision-making process because this step corresponds only to the failure detection phase. However, in future researches, during the implementation of the self-healing autonomic cycle, the people actor must play a role more active, as suggested by Flemisch et al. (2012). In such a case, the self-healing process will suggest the operator some fixes, and he/she will decide which of the suggestions fit the best to the current failure or propose a different solution. Moreover, the system can learn from the operator, in order to improve its recommendations (also, it is a topic of future researches).

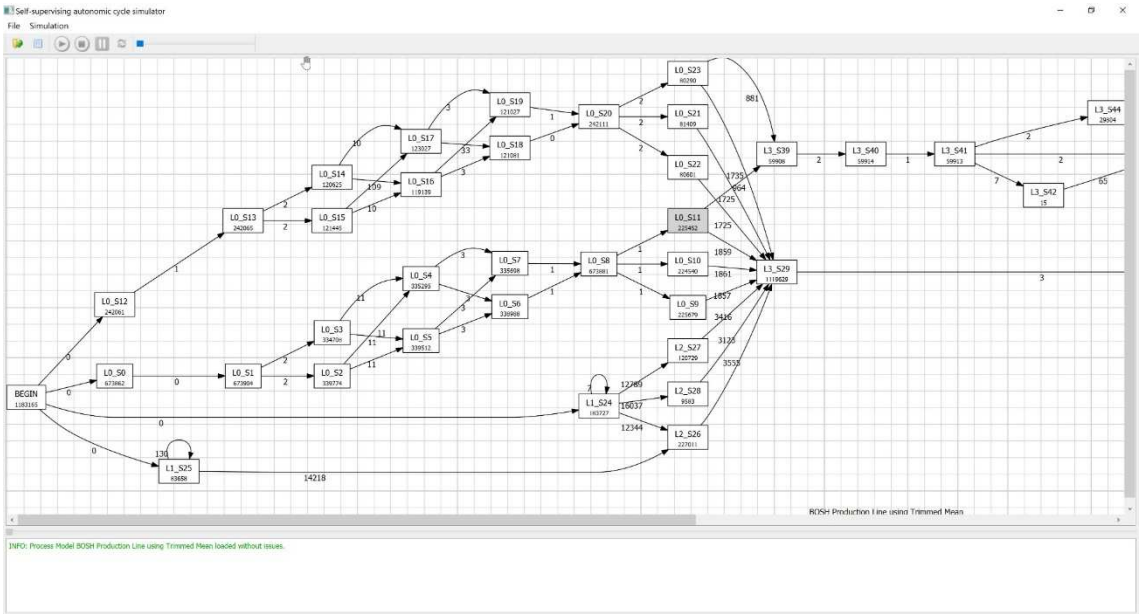

Figure 11. Self-supervising autonomic cycle prototype.

Fig 11 shows a screenshot of the prototype of the self-supervisory system. This prototype allows the operator to define the models must be used for failure detection and prediction (in our case, the process and predictive models), and to connect with the testing data, in order to replay each event of the manufacturing process and use our autonomic cycle for failure detection. The prototype allows the operator starting, stopping, pausing, restarting, and accelerating the test of the manufacturing process. Besides, this prototype diagnoses the system (predicts quality control test results and detect performance issues) and invokes the autonomic cycle autonomously for self-repairing when it is required. In this way, it implements our autonomic cycle of self-supervising.

5. Results and discussion

5.1. Results

In this subsection, we will discuss the result obtained in each phase of the supervisory system implementation.

5.1.1. Task 1

The process mining applied to the Bosch dataset allows us to build a process model for failure detection based on the following information.

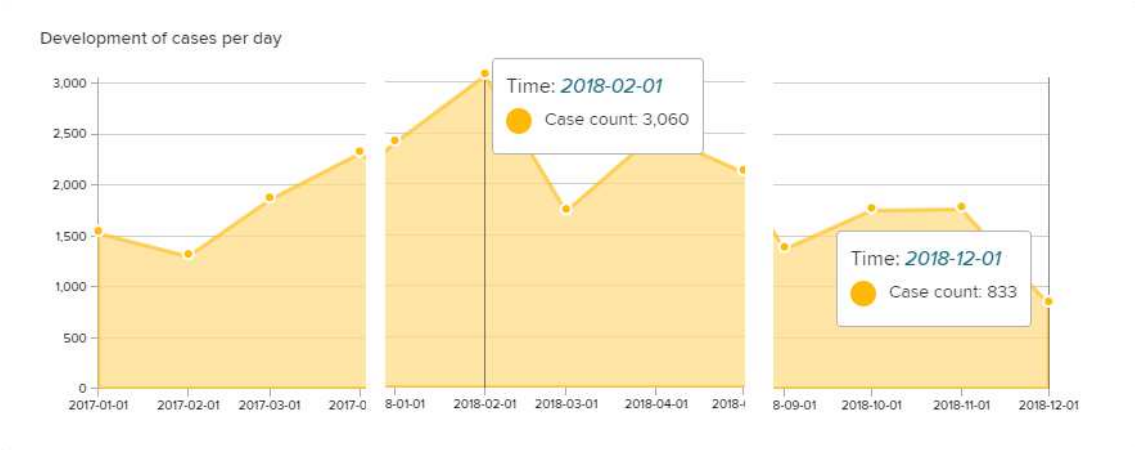

Figure 12. Min and max number of cases produced daily.

In the first place, 1779 parts are produced daily on average. Moreover, 833 parts are the minimum number of parts produced daily, and 3060 is the maximum (see Fig. 12). Another essential metric regarding the global performance of the manufacturing process is the average Throughput time (see Fig. 13), it indicates that each product is produced in environ 107 hours. It means that if during the production process execution, it is detected that this value is much higher than 107, then it represents a failure in the general performance of the production process.

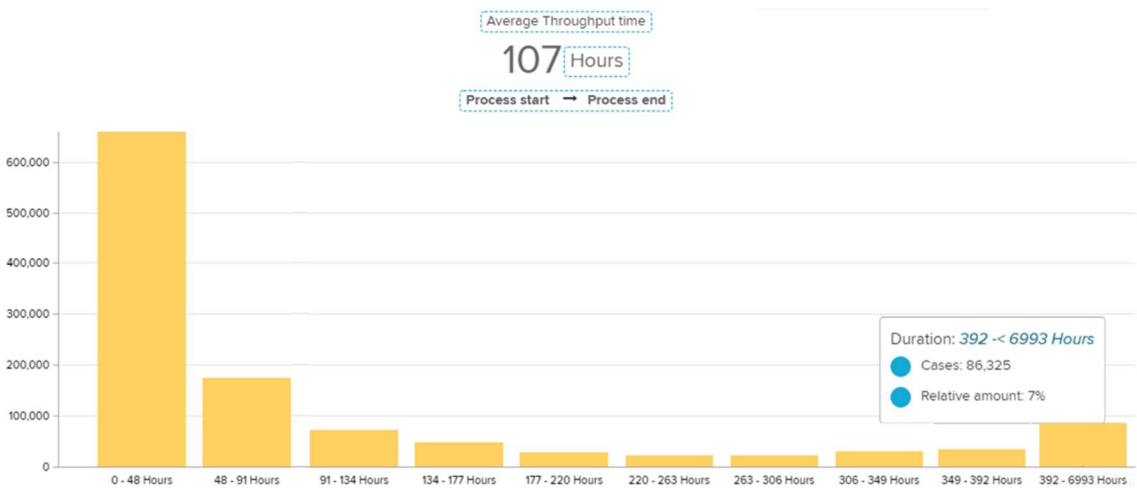

Figure 13. Throughput time.

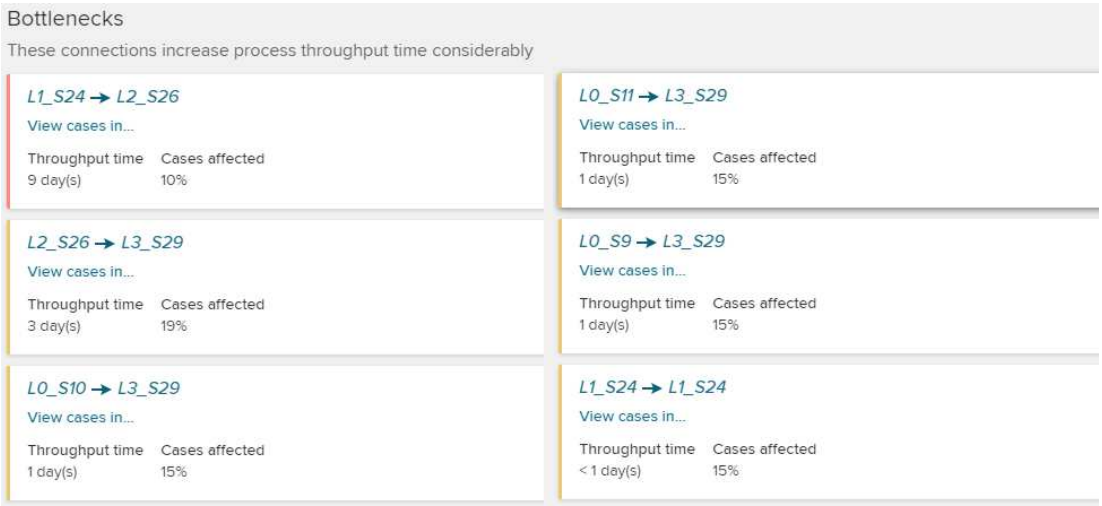

Figure 14. Stations with bottlenecks.

In Fig. 14, stations that present bottleneck are shown. E.g., station 24 can take until nine days processing one auto-part before passing it to station 26, affecting 10% of all the auto-parts produced. Similarly, the transition from station 26 to station 29 takes three days, which affects 19% of all the manufactured auto-parts. In general, the more significant bottlenecks are in transitions S24 to S26, S26 to S29, S10 to S29, S11 to S29, S9 to S29, and S24 to S24. This information is essential to detect failures in the execution of the manufacturing process actor's tasks, and let us know what stations need special attention.

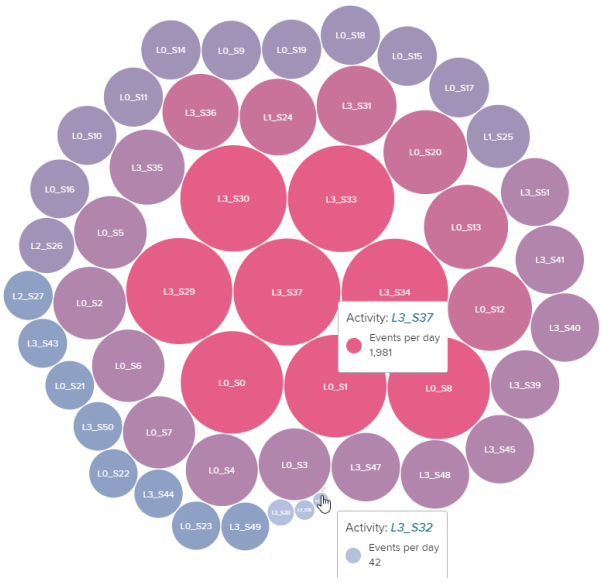

Figure 15. Stations workload.

Likewise, Fig. 15 shows the stations' workload. In this case, we can see that the station with the highest workload is station 37, with 1981 cases by day on average. Moreover, station 32 is the station with the lowest workload, only 42 cases by day. This statement confirms the insights that Mangal and Kumar (Mangal & Kumar, 2016) discover about station 32: "station 32 has the highest error rate. Also, station 32 does not process many products. Hence its impact on the production yield is minimal".

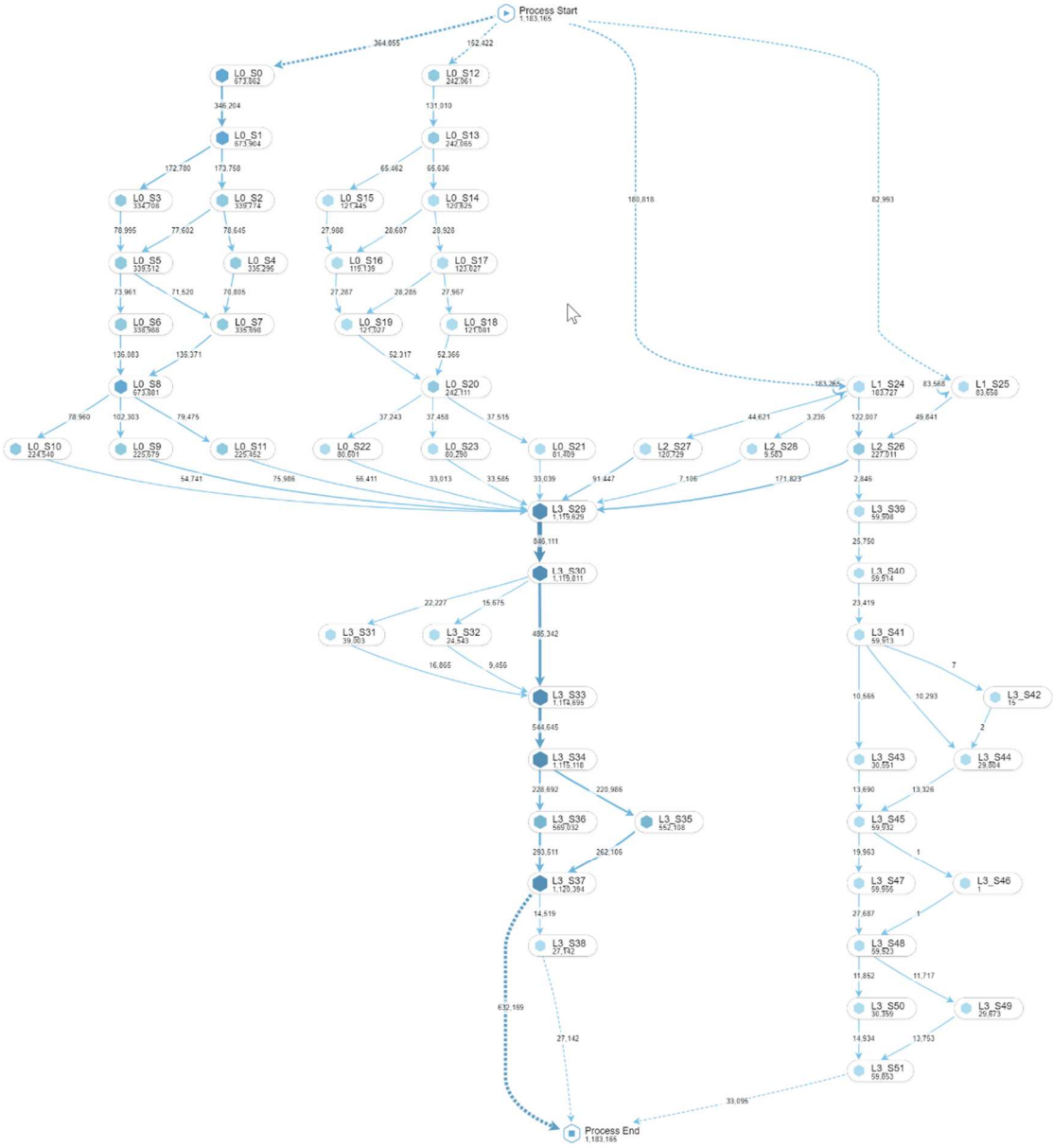

Figure 16. Bosch manufacturing process' flow.

Regarding the Bosch manufacturing process, Fig. 16 shows the layout of this production process, as well as the connections among actors. Each rounded rectangle represents one actor (a Bosch station). Lines represent transitions or paths (edges). This graph contains useful information about the stations' processing time (avg. time that each station takes to process an auto-part). This information is essential to detect failures at the stations' levels. Moreover, the prototype application uses this graph in order to know how each product is moving thought the production process.

From Fig. 16 can be deduced that although there are four production lines, they are not independent of each other.

5.1.2. Task 2

When a product is entering the manufacturing process, the predictive model is used to make quality control test predictions. Consequently, this prototype outputs information about failure prediction and detection in a dashboard, so that the operator can check what is happening in the manufacturing process (see Fig 17). The red circle represents the auto-parts being manufactured, and its movement thought the production line.

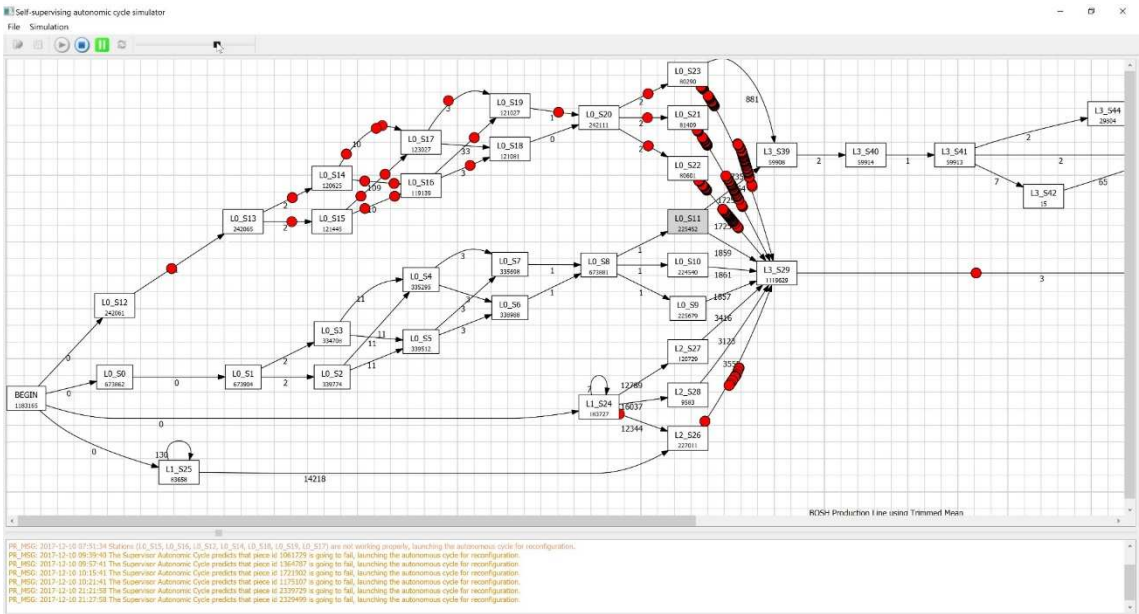

Figure 17. Bosch manufacturing process.

5.1.3. Task 3.

In essence, the autonomic cycle is continuously checking the performance of each station. If this performance is degraded respect to the value indicated by the process model, then the autonomic cycle of self-healing is launched. Similarly, when an auto-part finishes its production, this prototype compares the throughput time defined in the process model with the current throughput time, and if it is more significant respect to the time defined in the model, then the self-healing automatic cycle is launched. Fig. 18 shows the control messages about the failures and quality control test predictions.

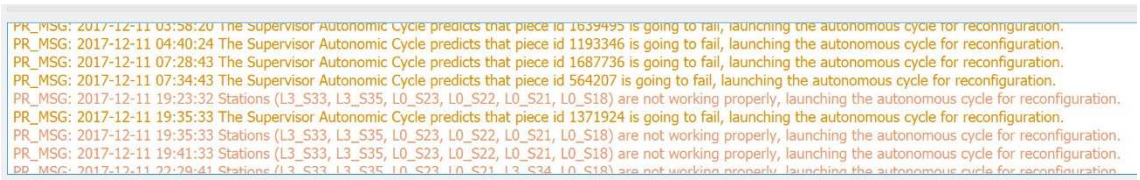

Figure 18. Self-supervision dashboard.

For instance, Fig. 18 shows failures and predictions. The first four message shows that some auto-part will fail the quality control test. Each message happens at different times of the day (during the execution of the manufacturing process), and it also shows that the autonomic cycle of self-healing was started. Similarly, messages 5 and 7-9, indicates that some stations are presenting failures because their performance was degraded, and the self-healing autonomic cycle was invoked.

We have measured the time (average) that this supervisory system takes to make a decision. It means, the average time that this supervisory system uses to predict if a product will fail or not the quality control test (performance of the classification model), as well as the time that the supervisory system employs to detect whether a station is working correctly or not (performance of the process mining model). Fig. 19 shows that those times have a low rate. It means that the supervisory system can quickly detect or

predict failures, which also is a desired characteristic in systems that require real-time processing (Lu, 2017).

Figure 19. Average time for failure and predictions.

Another important aspect of this supervisory system is that it can be generalized to other systems, by using the generic design presented in Section 2. However, the everything mining models must be appropriately built to represent the manufacturing process being studied. Essentially, our architecture can be used to promote self-* capabilities to any system. In part, due to the scalability of our system, and because this system is compatible with other reference architectures for Industry 4.0, like RAMI 4.0 (Pisching et al., 2018; Platform Industry 4.0, 2018), as is shown in Fig. 20.

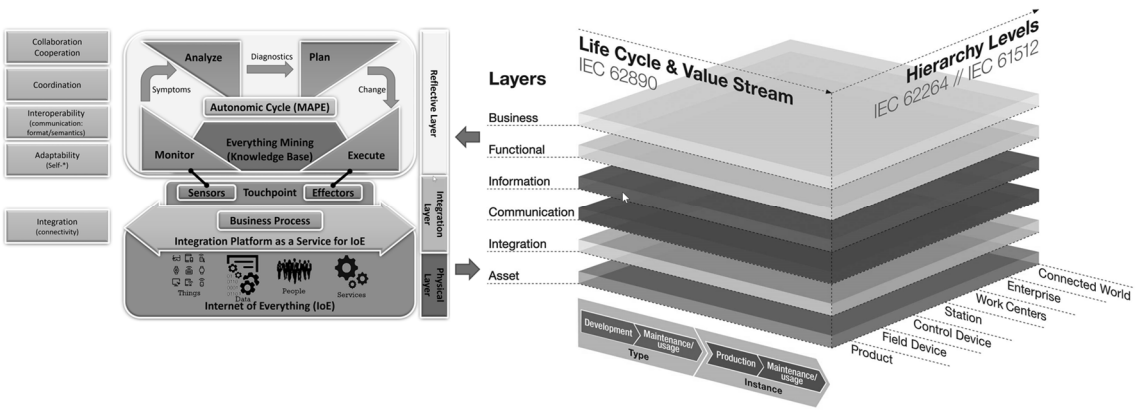

Figure 20. Compatibility AIFI and RAMI 4.0

AIFI 4.0 promotes the inclusion of self-* capabilities, like self-management, self-supervising, self-repairing, self-healing, etc. into RAMI 4.0, to promote the self-coordination, self-cooperation, and self-collaboration processes. Besides, the AIFI reflective layer brings support to the Information, Functional, and Business layers of RAMI. It means that the RAMI 4.0 layers use the knowledge and self-* properties deployed in AIFI, to endow autonomy in the manufacturing process. Moreover, The AIFI Physical and Integration layers are compatible with the Asset, Integration, and Communication layers of RAMI, respectively.

5.2. Verification of the quality criteria in our approach

In this subsection, a comparison with previous researches is made using the quality criteria defined in Section 2.

Concerning the *Number of Everything-mining techniques used by the supervisory system,* our self-supervising autonomic cycle uses two Everything-mining techniques. Properly, it has used process mining and data mining in order to create a process model and a predictive model that help in getting a diagnosis of the current status of the manufacturing process, and in initiating the corresponding autonomic cycle of self-healing if needed.

Regarding the *type of supervisory system that was built* (classification made by Xu et al. (2017)), our self-supervisory system is a value-driven supervisory system because it can detect failures that are not easily detected by traditional methods. In that sense, the supervisory system built on this research was tested with an anonymized dataset, but even when the dataset is anonymized, Tasks 1 and 2 of the autonomic cycle were able to get useful information to diagnose the manufacturing process, make quality control test predictions, and detect performance failures.

About the *use of any event-log Process-mining feature,* our research is focused on the use of the process mining technique, which gave us many insights about the manufacturing process, still when the dataset is anonymized. Notably, the process mining technique allows:

- Discovering the manufacturing process' layout.
- Discovering problematic stations and statistical data, like the throughput time and the average process time of each station, which was useful for failure detection.
- Detecting the relationship between actors and production lines. Besides, it detected that the production lines are not independent and that Line 0 consists of two sub-lines.

This information is not easy to get using other data analytics techniques. *This statement also proves that this supervisory system is a value-driven supervisory system*.

Relating to the *Studied Actors*, the architecture used in this research considers all the actors that can be involved in manufacturing processes. However, in this case study, we have worked with two actors:

- The *Process* actor was used in Task 1 to extract information about the manufacturing process, discover actors, relationships, production, and processing times, among others.
- The *Data* actor was used in Task 1 and Task 2. Task 1 got historical information about the stations' performance. Consequently, that information was useful for failure detection. Task 2 used the data generated by this actor to build the predictive model.

It is important to remark that we do not know which kind of actor represents each station because the Bosch dataset is anonymized and does not contain that information. Also, in the future, we expect to add support from other actors.

Concerning the *Scalability*, our architecture is scalable in many senses:

- Firstly, it accepts the inclusion of several everything-mining techniques.
- Secondly, all actors involved in manufacturing processes are considered as part of the architecture.
- Thirdly, this software was conceived to add incrementally other self-* properties to gain more and more autonomy with the inclusion of new data analytic autonomic cycles.
- Finally, the use of the Everything-mining techniques ensures the scalability of the system to treat any source of information.

Finally, regarding the *Autonomy of the supervisory system,* as can be seen from Fig. 8, 15, and 16, the system can make decisions for failure detection and quality control test predictions by itself (not human acts in the decision-making process to diagnose the system). In the future, this system will be able to allow not only self-supervising, but also self-planning and self-healing, with a focus on turning the manufacturing process into a fully coordinated, cooperative, and collaborative process in the context of Industry 4.0.

Unfortunately, we cannot establish any quantitative comparison with previous researches because most of them do not present quantitative results. Moreover, the few papers that show some results in numbers not have any metric that can be used for comparison.

6. Conclusions and Future Works

Industry 4.0 requires improving the levels of integrability and interoperability in a coordinated, cooperative, and collaborative way (Burns et al., 2019; Pietrewicz, 2019). This paper presented an architecture for autonomous integration and interoperability of actors in the Industry 4.0 context, which combines multiple paradigms to increase the autonomy of the manufacturing process. Moreover, this framework intends to increase the autonomy of a system by adding incrementally self-* properties.

The presented architecture serves as a support for other standard reference architectures for Industry 4.0, like RAMI 4.0 and IIRA (Lin et al., 2015). This research is not intending to replace the existing standards, but extends with solutions to deal with the challenges of Industry 4.0 not covered by the existing frameworks.

Specifically, this paper shows the implementation of the autonomic cycle of self-supervising as the first step towards self-coordinated manufacturing processes. The implementation of the autonomic cycle of self-supervising was detailed methodologically using MIDANO, allowing us to reach the desired results. It means that this autonomic cycle reached its designed goals of failure detection and quality control test predictions, intending to improve the autonomy of the manufacturing process.

Regarding the adaptability and scalability of this research to more complex systems, implies to join the system's experts with the data scientists in order to catch the data for the creating and validation of everything mining models, such as the manufacturing process model, the predictive models, and other models used to measure the devices' health. Particularly, this study was conducted over a vast and intricate manufacturing process, as the Bosh Production Line, which not only contains a considerable quantity of actors, but also generates a large quantity of imbalanced data. The results of this research work show that everything mining techniques are necessary to deal with the issue related to the self-organization of manufacturing processes.

Consequently, this study can serve as a guide to incorporate self-supervising properties to other manufacturing systems.

The results of this study allowed us to confirm the positive impact of combining the Autonomic computing paradigm, the Internet of Everything, and the Everything-mining, oriented to enable self-coordinated manufacturing processes in the context of Industry 4.0. Also, the main contributions of this research are:

- The capacity to support several Everything-mining techniques. Each mining technique improves the knowledge of the system, and by consequence, the decision-making processes.
- The integration of all actors involved in manufacturing processes, such as Thing, Data, People, and Services.
- The support for self-* properties that are added incrementally, guaranteeing, in this sense, the scalability of the system.
- The support for real-time analysis and decision-making.
- The capacity of diagnosing the system and to launch the self-healing autonomic cycle.

Future works are oriented to implement the autonomic cycles of self-configuring and self-healing, as well as the self-cooperation and self-collaboration autonomic managers, in order to turn manufacturing processes into a fully integrated and autonomous manufacturing system. Additionally, other features will be added to the prototype application developed in this research, so that it can support other Everything-mining techniques, in order to turn the manufacturing process into a smart and self-coordinated production system. Besides, in the future, we are planning to add one thing mining data analytical technique as DEKG, to get a proper failure diagnostic and increase

the number of failures that this system can detect. Concerning the scalability and adaptability of the self-supervising autonomic cycle presented in this paper to more complex use cases, it will be deeply addressed in future works.

Moreover, another future work is related to the utilization of our architecture in the real world. For that, we need to create a digital twin of all actors involved in the manufacturing process, so that each actor has its digital counterpart (Burns et al., 2019). Multi-agent systems can be useful for this task (Terán et al., 2017), or another platform, such as ROS Industrial (ROS Industrial, 2018). In this sense, a previous work conducted by Aguilar et al. (2017d) details how to deal with issues related to the bidirectional communication of software agents and cloud-services, naturally. Next, a message queue management system is essential to allow the autonomic cycles to receive the information or events needed to make decisions, using the existing everything-mining models. The digital counterpart of actors will be able to get information from the actor and activate its effectors so that they can act according to the orders given by the autonomic cycles, and change the environment according to the current requirements and context.

7. References.

Aguilar, J., Aguilar, K., Jerez, M., & Jiménez, C. (2017a). Implementación de tareas de analítica de datos para mejorar la calidad de servicios en redes de comunicaciones. *Publicaciones En Ciencias y Tecnología*, *11*(2), 63–77.

Aguilar, J., Buendia, O., Moreno, K., & Mosquera, D. (2016). Autonomous Cycle of Data Analysis Tasks for Learning Processes. *Technologies and Innovation*, 187–202. https://doi.org/10.1007/978-3-319-48024-4_15

Aguilar, J., Cordero, J., & Buendía, O. (2017b). Specification of the Autonomic Cycles of Learning Analytic Tasks for a Smart Classroom. *Journal of Educational Computing Research*, 0735633117727698. https://doi.org/10.1177/0735633117727698

Aguilar, J., Sanchez, M., Cordero, J., Valdiviezo-Díaz, P., Barba-Guamán, L., & Chamba-Eras, L. (2017c). Learning analytics tasks as services in smart

classrooms. *Universal Access in the Information Society*, *17*(4), 693–709. https://doi.org/10.1007/s10209-017-0525-0

Aguilar, J., Sanchez, M., Jerez, M., & Mendonca, M. (2017d). An Extension of the MiSCi Middleware for Smart Cities Based on Fog Computing. *Journal of Information Technology Research (JITR)*, *10*(4), 23–41. https://doi.org/10.4018/JITR.2017100102

Berdal, Q., Pacaux-Lemoine, M.-P., Trentesaux, D., & Chauvin, C. (2019). Human-Machine Cooperation in Self-organized Production Systems: A Point of View. In T. Borangiu, D. Trentesaux, A. Thomas, & S. Cavalieri (Eds.), *Service Orientation in Holonic and Multi-Agent Manufacturing* (pp. 123–132). Springer International Publishing. https://doi.org/10.1007/978-3-030-03003-2_9

Birkel, H. S., Veile, J. W., Müller, J. M., Hartmann, E., & Voigt, K.-I. (2019). Development of a Risk Framework for Industry 4.0 in the Context of Sustainability for Established Manufacturers. *Sustainability*, *11*(2), 384. https://doi.org/10.3390/su11020384

Breiman, L. (1996). Bagging Predictors. *Machine Learning*, *24*(2), 123–140. https://doi.org/10.1023/A:1018054314350

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., Vanderplas, J., Joly, A., Holt, B., & Varoquaux, G. (2013). API design for machine learning software: Experiences from the scikit-learn project. *ArXiv:1309.0238 [Cs]*. http://arxiv.org/abs/1309.0238

Burns, T., Cosgrove, J., & Doyle, F. (2019). A Review of Interoperability Standards for Industry 4.0. *Procedia Manufacturing*, *38*, 646–653. https://doi.org/10.1016/j.promfg.2020.01.083

Cao, Q., Giustozzi, F., Zanni-Merk, C., Beuvron, F., & Reich, C. (2019). Smart Condition Monitoring for Industry 4.0 Manufacturing Processes: An Ontology-Based Approach. *Cybernetics and Systems*, *50*, 1–15. https://doi.org/10.1080/01969722.2019.1565118

Celonis SE. (2019). *Celonis*. Celonis. https://www.celonis.com/

Collier, J. (2002). What is Autonomy? *International Journal of Computing Anticipatory Systems: CASY 2001 - Fifth International Conference.*, *20*. http://cogprints.org/2289/

Dencker, K., Fasth, Å., Stahre, J., Mårtensson, L., Lundholm, T., & Akillioglu, H. (2009). Proactive assembly systems-realising the potential of human collaboration with automation. *Annual Reviews in Control*, *33*(2), 230–237. https://doi.org/10.1016/j.arcontrol.2009.05.004

Derboul, A., Hadj, B. I., & Chafik, K. (2018). Contribution of Industrial Information Systems to Industrial Performance: Case of Industrial Supervision. *Springer International Publishing AG, Part of Springer Nature*, 884–901. https://doi.org/10.1007/978-3-319-74500-8_79

Dinardo, G., Fabbiano, L., & Vacca, G. (2018). A smart and intuitive machine condition monitoring in the Industry 4.0 scenario. *Measurement*, *126*, 1–12. https://doi.org/10.1016/j.measurement.2018.05.041

Elattar, M., Wendt, V., & Jasperneite, J. (2017). Communications for Cyber-Physical Systems. In *Industrial Internet of Things* (pp. 347–372). Springer, Cham. https://doi.org/10.1007/978-3-319-42559-7_13

Flemisch, F., Heesen, M., Hesse, T., Kelsch, J., Schieben, A., & Beller, J. (2012). Towards a dynamic balance between humans and automation: Authority, ability, responsibility and control in shared and cooperative control situations. *Cognition, Technology & Work*, *14*(1), 3–18. https://doi.org/10.1007/s10111-011-0191-6

Gaham, M., Bouzouia, B., & Achour, N. (2015). Human-in-the-Loop Cyber-Physical Production Systems Control (HiLCP2sC): A Multi-objective Interactive Framework Proposal. In T. Borangiu, A. Thomas, & D. Trentesaux (Eds.), *Service Orientation in Holonic and Multi-agent Manufacturing* (pp. 315–325). Springer International Publishing. https://doi.org/10.1007/978-3-319-15159-5_29

Haupert, J., Klinge, X., & Blocher, A. (2017). CPS-Based Manufacturing with Semantic Object Memories and Service Orchestration for Industrie 4.0 Applications. In *Industrial Internet of Things* (pp. 203–229). Springer, Cham. https://doi.org/10.1007/978-3-319-42559-7_8

IBM. (2004). *Autonomic Computing User's Guide*. https://www.ibm.com/developerworks/autonomic/books/fpu0mst.htm

Kaggle. (2016). *Bosch Production Line Performance*. https://kaggle.com/c/bosch-production-line-performance

Kalatzis, N., Routis, G., Marinellis, Y., Avgeris, M., Roussaki, I., Papavassiliou, S., & Anagnostou, M. (2019). Semantic Interoperability for IoT Platforms in Support

of Decision Making: An Experiment on Early Wildfire Detection. *Sensors (Basel, Switzerland)*, *19*(3). https://doi.org/10.3390/s19030528

Kumar, A., Glisson, W., & Cho, H. (2020, January 7). *Network Attack Detection Using an Unsupervised Machine Learning Algorithm*. https://doi.org/10.24251/HICSS.2020.795

Lalanda, P., McCann, J. A., & Diaconescu, A. (2013). *Autonomic Computing*. Springer London. https://doi.org/10.1007/978-1-4471-5007-7

Lee, J., Ardakani, H. D., Yang, S., & Bagheri, B. (2015). Industrial Big Data Analytics and Cyber-physical Systems for Future Maintenance & Service Innovation. *Procedia CIRP*, *38*, 3–7. https://doi.org/10.1016/j.procir.2015.08.026

Lee, J., Kao, H.-A., & Yang, S. (2014). Service Innovation and Smart Analytics for Industry 4.0 and Big Data Environment. *Procedia CIRP*, *16*, 3–8. https://doi.org/10.1016/j.procir.2014.02.001

Lemaitre, G., Oliveira, D., & Aridas, C. (2014a). *Balanced Bagging Classifier*. Imbalanced Learn. https://imbalanced-learn.readthedocs.io/en/stable/generated/imblearn.ensemble.BalancedBaggingClassifier.html

Lemaitre, G., Oliveira, D., & Aridas, C. (2014b). *Balanced Random Forest Classifier*. Imbalanced Learn. https://imbalanced-learn.readthedocs.io/en/stable/generated/imblearn.ensemble.BalancedRandomForestClassifier.html

Leżański, P. (2017). Architecture of Supervisory Systems For Subtractive Manufacturing Processes In Industry 4.0 Based Manufacturing. *Journal of Machine Construction and Maintenance*, *104*, 59–64.

Li, D., Tang, H., Wang, S., & Liu, C. (2017). A big data enabled load-balancing control for smart manufacturing of Industry 4.0. *Cluster Computing*, *20*(2), 1855–1864. https://doi.org/10.1007/s10586-017-0852-1

Liao, Y., Ramos, L. F. P., Saturno, M., Deschamps, F., de Freitas Rocha Loures, E., & Szejka, A. L. (2017). The Role of Interoperability in The Fourth Industrial Revolution Era. *IFAC-PapersOnLine*, *50*(1), 12434–12439. https://doi.org/10.1016/j.ifacol.2017.08.1248

Lin, S.-W., Miller, B., Durand, J., Joshi, R., Didier, P., Amine, C., Torenbeek, R., Duggal, D., Martin, R., Bleakleay, G., & others. (2015). *Industrial internet reference architecture* (Tech. Rep No. 2017-01–25). Industrial Internet Consortium (IIC).

https://www.iiconsortium.org/pdf/SHI-WAN%20LIN_IIRA-v1%208-release-20170125.pdf

Liu, Y., Yuen, C., Cao, X., Hassan, N. U., & Chen, J. (2014). Design of a Scalable Hybrid MAC Protocol for Heterogeneous M2M Networks. *IEEE Internet of Things Journal*, *1*(1), 99–111. https://doi.org/10.1109/JIOT.2014.2310425

Lu, Y. (2017). Industry 4.0: A survey on technologies, applications and open research issues. *Journal of Industrial Information Integration*, *6*, 1–10. https://doi.org/10.1016/j.jii.2017.04.005

Mangal, A., & Kumar, N. (2016). Using big data to enhance the bosch production line performance: A Kaggle challenge. *2016 IEEE International Conference on Big Data (Big Data)*, 2029–2035. https://doi.org/10.1109/BigData.2016.7840826

Martino, B. D., Li, K.-C., Yang, L. T., & Esposito, A. (2018). Trends and Strategic Researches in Internet of Everything. In *Internet of Everything* (pp. 1–12). Springer, Singapore. https://doi.org/10.1007/978-981-10-5861-5_1

Morris, W. (Ed.). (1982). *The American Heritage dictionary* (2nd college ed). Houghton Mifflin.

Obitko, M., & Jirkovský, V. (2015). Big Data Semantics in Industry 4.0. *Industrial Applications of Holonic and Multi-Agent Systems*, 217–229. https://doi.org/10.1007/978-3-319-22867-9_19

O'Brien, R., & Ishwaran, H. (2019). A random forests quantile classifier for class imbalanced data. *Pattern Recognition*, *90*, 232–249. https://doi.org/10.1016/j.patcog.2019.01.036

Oesterreich, T. D., & Teuteberg, F. (2016). Understanding the implications of digitisation and automation in the context of Industry 4.0: A triangulation approach and elements of a research agenda for the construction industry. *Computers in Industry*, *83*, 121–139. https://doi.org/10.1016/j.compind.2016.09.006

Pacaux-Lemoine, M.-P., Trentesaux, D., Zambrano Rey, G., & Millot, P. (2017). Designing intelligent manufacturing systems through Human-Machine Cooperation principles: A human-centered approach. *Computers & Industrial Engineering*, *111*, 581–595. https://doi.org/10.1016/j.cie.2017.05.014

Pacheco, F., Aguilar, J., Rangel, C., Cerrada, M., & Altamiranda, J. (2014). Methodological framework for data Processing based on the data science paradigm. *Computing Conference (CLEI), XL Latin American*, 1–12.

Pal, M. (2005). Random forest classifier for remote sensing classification. *International Journal of Remote Sensing*, *26*(1), 217–222. https://doi.org/10.1080/01431160412331269698

Parashar, M., & Hariri, S. (2005). Autonomic Computing: An Overview. In J.-P. Banâtre, P. Fradet, J.-L. Giavitto, & O. Michel (Eds.), *Unconventional Programming Paradigms* (pp. 257–269). Springer Berlin Heidelberg.

Pedone, G., & Mezgár, I. (2018). Model similarity evidence and interoperability affinity in cloud-ready Industry 4.0 technologies. *Computers in Industry*, *100*, 278–286. https://doi.org/10.1016/j.compind.2018.05.003

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825−2830.

Pietrewicz, L. (2019). Coordination in the age of Industry 4.0. *Economic and Social Development: Book of Proceedings*, 264–274.

Pisching, M. A., Pessoa, M. A. O., Junqueira, F., dos Santos Filho, D. J., & Miyagi, P. E. (2018). An architecture based on RAMI 4.0 to discover equipment to process operations required by products. *Computers & Industrial Engineering*, *125*, 574–591. https://doi.org/10.1016/j.cie.2017.12.029

Platform Industry 4.0. (2018). *Reference Architectural Model Industrie 4.0 (RAMI4.0)—An Introduction*. Platform Industry 4.0. https://www.plattform-i40.de/PI40/Redaktion/EN/Downloads/Publikation/rami40-an-introduction.html

Python TM. (2019). *Welcome to Python.org*. Python.Org. https://www.python.org/

Rangel, C., Pacheco, F., Aguilar, J., & Cerrada, M. (2013). Methodology for detecting the feasibility of using data mining in an organization. *Computing Conference (CLEI), XXXIX Latin American*, 502–513.

Reis, M., & Gins, G. (2017). Industrial Process Monitoring in the Big Data/Industry 4.0 Era: From Detection, to Diagnosis, to Prognosis. *Processes*, *5*(35), 1–16. https://doi.org/10.3390/pr5030035

Rojas, R. A., Rauch, E., Vidoni, R., & Matt, D. T. (2017). Enabling Connectivity of Cyber-physical Production Systems: A Conceptual Framework. *Procedia Manufacturing*, *11*, 822–829. https://doi.org/10.1016/j.promfg.2017.07.184

Romero, D., Bernus, P., Noran, O., Stahre, J., & Fast-Berglund, Å. (2016). The Operator 4.0: Human Cyber-Physical Systems & Adaptive Automation Towards Human-Automation Symbiosis Work Systems. In I. Nääs, O. Vendrametto, J. Mendes Reis, R. F. Gonçalves, M. T. Silva, G. von Cieminski, & D. Kiritsis (Eds.), *Advances in Production Management Systems. Initiatives for a Sustainable World* (pp. 677–686). Springer International Publishing. https://doi.org/10.1007/978-3-319-51133-7_80

ROS Industrial. (2018, November 16). *ROS-Industrial*. ROS Industrial. https://rosindustrial.org/

Sanchez, M., Aguilar, J., & Exposito, E. (2018a). Fog computing for the integration of agents and web services in an autonomic reflexive middleware. *Software Oriented Computing and Applications*, *12*(333), 1–15.

Sanchez, M., Aguilar, J., & Exposito, E. (2018b). Integración SOA-MAS en Ambientes Inteligentes. *DYNA*, *85*(206), 268–282. https://doi.org/10.15446/dyna.v85n206.68671

Sanchez, M., Exposito, E., & Aguilar, J. (2019a). Autonomic Cycles of Everything Mining for Coordination in the Context of the Industry 4.0. *submitted to publication, Journal of Industrial Information Integration, In Review*.

Sanchez, M., Exposito, E., & Aguilar, J. (2019b). Industry 4.0 Survey from a System Integration Perspective. *Submitted to Publication, Journal of Computer Integration Manufacturing, In Review*.

Sanchez, M., Exposito, E., & Aguilar, J. (2019c). *Self-Supervising Autonomic cycle replayer*. https://bitbucket.org/mbsanchez/ssacsimulator/src/master/DataSets%20I4.0/PreProcess/

Santos, M. Y., Sá, J. O. e, Costa, C., Galvão, J., Andrade, C., Martinho, B., Lima, F. V., & Costa, E. (2017). A Big Data Analytics Architecture for Industry 4.0. *Recent Advances in Information Systems and Technologies*, 175–184. https://doi.org/10.1007/978-3-319-56538-5_19

Silva, R., Rocha, A. D., Leitao, P., & Barata, J. (2018). IDARTS – Towards intelligent data analysis and real-time supervision for industry 4.0. *Computers in Industry*, *101*, 138–146. https://doi.org/10.1016/j.compind.2018.07.004

Singh, Y., Kaur, A., & Malhotra, R. (2009). Empirical validation of object-oriented metrics for predicting fault proneness models. *Software Quality Journal*, *18*(1), 3. https://doi.org/10.1007/s11219-009-9079-6

Singla, L., & Agrawal, P. (2016). *Bosch Production Line Performance* [PDF]. http://neddimitrov.org/uploads/classes/201604CO/LukeshPrateek-BoschFailurePrediction.pdf

Sterritt, R., & Hinchey, M. (2005). Autonomic Computing " Panacea or Poppycock? *Proceedings of the 12th IEEE International Conference and Workshops on Engineering of Computer-Based Systems*, 535–539. https://doi.org/10.1109/ECBS.2005.22

Suali, A. J., Fauzi, S. S. M., Nasir, M. H. N. M., & Sobri, W. a. W. M. (2017). Synthesizing the Literature on the Issues of Coordination and its Impact on Software Quality. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, *9*(3–3), 27–31.

Terán, J., Aguilar, J., & Cerrada, M. (2017). Integration in industrial automation based on multi-agent systems using cultural algorithms for optimizing the coordination mechanisms. *Computers in Industry*, *91*, 11–23. https://doi.org/10.1016/j.compind.2017.05.002

Tharwat, A., Gaber, T., Ibrahim, A., & Hassanien, A. E. (2017). Linear discriminant analysis: A detailed tutorial. *AI Communications*, *30*(2), 169–190. https://doi.org/10.3233/AIC-170729

The-Qt-Company. (2019). *Qt-Cross-platform software development for embedded & desktop*. https://www.qt.io

Tiboni, M., Aggogeri, F., Pellegrini, N., & Perani, C. A. (2019). Smart Modular Architecture for Supervision and Monitoring of a 4.0 Production Plant. *Int. J. of Automation Technology*, *13*(2), 310–318.

Truszkowski, W., Hallock, H. L., Rouff, C., Karlin, J., Rash, J., Hinchey, M., & Sterritt, R. (2010). Introduction. In W. Truszkowski, H. Hallock, C. Rouff, J. Karlin, J. Rash, M. Hinchey, & R. Sterritt (Eds.), *Autonomous and Autonomic Systems: With Applications to NASA Intelligent Spacecraft Operations and Exploration Systems: With Applications to NASA Intelligent Spacecraft Operations and Exploration Systems* (pp. 3–23). Springer London. https://doi.org/10.1007/b105417_1

Vizcarrondo, J., Aguilar, J., Exposito, E., & Subias, A. (2017). MAPE-K as a service-oriented architecture. *IEEE Latin America Transactions*, *15*(6), 1163–1175. https://doi.org/10.1109/TLA.2017.7932705

Vizcarrondo, J., Aguilar, J., Exposito, E., & Subias, A. (2012). ARMISCOM: Autonomic reflective middleware for management service composition. *2012 Global Information Infrastructure and Networking Symposium (GIIS)*, 1–8. https://doi.org/10.1109/GIIS.2012.6466760

Wang, Y., Sheng, Y., Wang, J., & Zhang, W. (2017, November 14). *Human Intention Estimation With Tactile Sensors in Human-Robot Collaboration*. ASME 2017 Dynamic Systems and Control Conference. https://doi.org/10.1115/DSCC2017-5291

Wardhani, N. W. S., Rochayani, M. Y., Iriany, A., Sulistyono, A. D., & Lestantyo, P. (2019). Cross-validation Metrics for Evaluating Classification Performance on Imbalanced Data. *2019 International Conference on Computer, Control, Informatics and Its Applications (IC3INA)*, 14–18. https://doi.org/10.1109/IC3INA48034.2019.8949568

Xu, L., He, W., & Li, S. (2014). Internet of Things in Industries: A Survey. *IEEE Transactions on Industrial Informatics*, *10*, 2233–2243. https://doi.org/10.1109/TII.2014.2300753

Xu, Y., Sun, Y., Wan, J., Liu, X., & Song, Z. (2017). Industrial Big Data for Fault Diagnosis: Taxonomy, Review, and Applications. *IEEE Access*, *5*, 138–146. https://doi.org/10.1109/ACCESS.2017.2731945

Yang, L. T., Di Martino, B., & Zhang, Q. (2017). Internet of Everything. *Mobile Information Systems*, *2017*, 1–3. https://doi.org/10.1155/2017/8035421

Yang, S.-W., & Chen, Y.-K. (2013). The M2M Connectivity Framework: Towards an IoT Landscape. *2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing*, 572–579. https://doi.org/10.1109/GreenCom-iThings-CPSCom.2013.108

Zhou, X., Xu, Z., Wang, L., & Chen, K. (2017). What should we do? A structured review of SCADA system cyber security standards. *2017 4th International Conference on Control, Decision and Information Technologies (CoDIT)*, 0605–0614. https://doi.org/10.1109/CoDIT.2017.8102661