Autonomic process management for Integration in Industry 4.0

Manuel Sanchez

École doctorale des sciences exactes et leurs applications
Doctorado en ciencias aplicadas de la Universidad de Los Andes

Autonomic Process Management for Industry 4.0

THÈSE

pour l'obtention du

Doctorat de l'Université de Pau et des Pays de l'Adour (France)
(mention Informatique)

et

Doctor por la Universidad de Los Andes (Venezuela)
(Doctor en ciencias aplicadas)

par

Manuel Baldemar SANCHEZ

Sutenue le 29 Mai 2020 devant le jury composé de:

Rapporteurs:	
Audine SUBIAS	INSA Toulouse
José GUTIERREZ	Université d'Alcalá, Espagne
Examinateurs:	
Myriam LAMOLLE	Univ Pau & Pays Adour
Directeurs:	
Ernesto EXPOSITO	Univ Pau & Pays Adour
Jose AGUILAR	Universidad de Los Andes

Laboratoire d'informatique de l'Université de Pau et des Pays de L'Adour (LIUPPA)
Centro de Estudios en Microelectrónica y Sistemas (CEMISID)

Acknowledgments

First, I would like to thank God because he guides my steps each day, and gives me the health and the strength to get ahead, with God everything is made, and nothing is made except through him.

Also, I want to thank my family (brothers, cousins, uncles, nephews, nieces, brothers-in-law, parents-in-law, parents, and the others), especially to my mother, who teaches me that study is the only way to grow in many senses, break barriers and to make great things. Thanks, mom, you must be very proud of me from the Sky.

Special thanks to my wife, Sonia, and to my children Yeray and Angela, for always being there with patience and love in good and bad times. Without you, life has no sense.

I also want to express my most enormous thanks to my two supervisors Jose Aguilar and Ernesto Exposito, for allowing me to do this Ph.D. and for guiding me during this work. I appreciate their tolerance and collaboration very much.

I am very grateful to all my friends and the great people that I came across during the period of my thesis.

Abstract

Because of the digital revolution, also known as Industry 3.0, the boundaries between the physical and digital worlds are shrinking to give life to a more interconnected and smart factories. These factories allow employees, machines, processes, and products to interact oriented to provide a better organization of all the productive means, empowering the entire company itself to achieve higher levels of efficiency and productivity. These technologies are profoundly transforming our society, allowing customizing everything in detail, reducing goods and services costs, transforming worker's and job's conditions for safety and security, among others. In that sense, Industry 3.0 acted as a catalyst that promoted new production mechanisms, which originated a new industrial revolution known as Industry 4.0. The concept of Industry 4.0, is used to designate the new generation of connected, robotics, and intelligent factories. Fundamentally, the vision of Industry 4.0 is to give smart capabilities to the production and physical operations to create a more holistic and better-connected ecosystem.

One crucial aspect to consider, regarding the idea of the Industry 4.0 concept, is related to integrability and interoperability of the actors involved in manufacturing processes. It means that people, things, processes, and data have to be able not only to make decisions for themselves and to carry out their work in a more autonomous way (independence) but, also, the self-management of the whole factory (need to promote integrability and interoperability). The previous statement implies that the production processes' actors should be able to autonomously negotiate in order to reach agreements linked to achieve both individual and collective production goals. In that sense, Industry 4.0 represents not only a new way to produce goods and services but also a crucial integration challenge of the actors involved in the manufacturing processes that need connection, communication, coordination, cooperation, and collaboration (denoted as 5C) capabilities that allow them to comply with the vision of Industry 4.0.

Principally, this thesis aims at empowering processes management for Industry 4.0, proposing a stack of five levels, denoted as 5C. The 5C stack levels represent a way to deal with integration and interoperability challenges so that they can be solved incrementally at each level. From this perspective, we must start solving connection and communication issues as a first step to promote more elaborated organization processes like coordination, cooperation, and collaboration. Mainly, the 5C denote the elements needed to allow autonomous integration and interoperability of actors in Industry 4.0.

From this point of view, in this thesis project, we present a first contribution that is oriented to deal with the integration challenges regarding the Industry 4.0 context at the level of connection and communication. This solution is based in a Multi-agent system in which the physical elements of the system are characterized virtually as agents. Notably, the use of Multi-agent systems allows creating an intelligent environment dotted with characteristics of autonomy, decentralization, self-organization, self-direction, standardized protocol, and other properties of Multi-agent systems. Moreover, the proposed solution allows actors to extend their limited

capabilities with service deployed through the Internet, as an intent to automatize, optimize, and in more mature stages, transform any environment into a fully integrated, automated, and intelligent environment. Consequently, the proposed architecture will be evaluated and compared to previous researches in this field.

In the second place, we will solve some integration challenges of Industry 4.0 at the level of coordination, cooperation, and collaboration. In this case, we design a framework for autonomous integration of actors in Industry 4.0, to allow them to autonomously coordinate, cooperate, and collaborate. This framework uses technologies like the Internet of Everything, Everything mining, and Autonomic computing. Next, we design some autonomic cycles of data analytics tasks, oriented to enable autonomous coordination in manufacturing processes. Fundamentally, these data analytics tasks create the knowledge bases needed in a production environment to support self-planning, self-manage, self-supervising, self-healing, etc. to the manufacturing process.

Finally, we implement an autonomous cycle of data analytics tasks for self-supervising, using several Everything-mining techniques over data sources corresponding to a real manufacturing process. It defines a self-value-driven supervisory system, according to the classification made by Xu et al. (2017), that can process and verify the functionalities and applicability of our framework in manufacturing processes. Moreover, the self-supervising system developed in this thesis project is compared to other research works.

Keywords: Industry 4.0; Integration; Interoperability; Multi-agent systems; Service Oriented Architecture; Internet of Everything; Autonomic computing; Everything mining.

Résumé

En raison de la révolution numérique, également connue sous le nom d'Industrie 3.0, les frontières entre les mondes physique et digital se rétrécissent pour donner vie à des usines plus interconnectées et intelligentes dans lesquelles les employés, les machines, les processus et les produits interagissent de manière à donner une meilleure organisation de tous les moyens productifs, toute l'entreprise elle-même pour atteindre des niveaux plus élevés d'efficacité et de productivité. Ces technologies transforment profondément notre société, permettant de tout personnaliser en détail, de réduire les coûts des biens et services, de transformer les conditions de sécurité des travailleurs, entre autres. En ce sens, l'Industrie 3.0 a agi comme un catalyseur qui a promu de nouveaux mécanismes de production, à l'origine d'une nouvelle révolution industrielle connue sous le nom d'Industrie 4.0. Le concept d'Industrie 4.0 est utilisé pour désigner la nouvelle génération d'usines connectées, robotiques et intelligentes. Fondamentalement, la vision de l'Industrie 4.0 est de donner des capacités intelligentes à la production et aux opérations physiques pour créer un écosystème plus holistique et mieux connecté.

Un aspect crucial à considérer, concernant l'idée du concept de l'Industrie 4.0, est lié à l'intégrabilité et à l'interopérabilité des acteurs impliqués dans les processus de fabrication. Cela signifie que les personnes, les dispositifs, les processus et les données doivent être capables non seulement de prendre des décisions pour eux-mêmes et d'effectuer leur travail de la manière la plus autonome (indépendance), mais aussi de promouvoir l'autogestion de l'ensemble de l'usine (nécessité de promouvoir l'intégrabilité et l'interopérabilité). La déclaration précédente implique que les acteurs des processus de production doivent pouvoir négocier de manière autonome afin de parvenir à des accords liés à la réalisation d'objectifs de production individuels et collectifs. En ce sens, l'Industrie 4.0 représente non seulement une nouvelle façon de produire des biens et des services, mais aussi, un défi d'intégration crucial des acteurs impliqués dans les processus de fabrication qui ont besoin de capacités de connexion, de communication, de coordination, de coopération et de collaboration (notées 5C) qui leur permettent de se conformer à la vision de l'industrie 4.0.

Principalement, cette thèse vise à responsabiliser la gestion des processus pour l'Industrie 4.0, en proposant une pile de cinq niveaux, notée 5C. Les niveaux de pile 5C représentent un moyen de relever les défis d'intégration et d'interopérabilité afin qu'ils puissent être résolus de manière incrémentielle à chaque niveau. Dans cette perspective, nous devons commencer à résoudre les problèmes de connexion et de communication comme une première étape pour promouvoir des processus d'organisation plus élaborés comme la coordination, la coopération et la collaboration. Essentiellement, le 5C désigne des éléments essentiels pour permettre l'intégration autonome et l'interopérabilité des acteurs de l'Industrie 4.0.

De ce point de vue, dans ce projet de thèse, nous présentons une première contribution qui est destinée à faire face aux défis d'intégration du contexte de l'Industrie 4.0 au niveau de la connexion et de la communication. Cette solution est basée sur un système multi-agents dans lequel les éléments physiques du système sont caractérisés virtuellement comme des agents. Notamment, l'utilisation de systèmes

multi-agents permet de créer un environnement intelligent parsemé de caractéristiques d'autonomie, de décentralisation, d'auto-organisation, d'auto-direction, de protocole normalisé et d'autres propriétés des systèmes multi-agents. De plus, la solution proposée permet aux acteurs d'étendre leurs capacités limitées avec un service déployé via Internet, dans le but d'automatiser, d'optimiser et, à des stades plus matures, de transformer n'importe quel environnement en un environnement entièrement intégré, automatisé et intelligent. Par conséquent, l'architecture proposée sera évaluée et comparée aux recherches menées dans ce domaine.

En second lieu, nous allons résoudre certains défis d'intégration de l'Industrie 4.0 au niveau de la coordination, de la coopération et de la collaboration. Dans ce cas, nous concevons un cadre d'intégration autonome des acteurs de l'Industrie 4.0, pour leur permettre de se coordonner, de coopérer et de collaborer de manière autonome. Ce cadre utilise des technologies comme l'Internet de tous, l'exploration minière de tous et l'informatique autonome. Ensuite, nous concevons des cycles autonomes de tâches d'analyse de données, orientés pour permettre une coordination autonome dans les processus de fabrication. Fondamentalement, ces tâches d'analyse de données créent les bases de connaissances nécessaires dans un environnement de production pour prendre en charge l'auto-planification, l'autogestion, l'auto-supervision, l'auto-guérison, etc. au processus de fabrication.

Enfin, nous mettons en œuvre un cycle autonome de tâches d'analyse de données pour l'autosurveillance, en utilisant plusieurs techniques d'exploration minière de tout sur des sources de données correspondant à un véritable processus de fabrication. Il définit un système de supervision basé sur la valeur propre, selon la classification faite par Xu et al. (2017), qui peut traiter et vérifier les fonctionnalités et l'applicabilité de notre cadre dans les processus de fabrication. De plus, le système d'auto-supervision développé dans ce projet de thèse sera comparé à d'autres travaux de recherche.

Mots-clés: Industrie 4.0; Intégration; Interopérabilité; Systèmes multi-agents; Architecture orientée services; Internet of Everything; Informatique autonomique; Everything mining.

Publications

Journal

- M. Sanchez, J. Aguilar, E. Exposito. "Fog computing for the integration of agents and web services in an autonomic reflexive middleware". Service-Oriented Computing and Applications (Q3), pp. 1-15. 2018. https://goo.gl/eCyVqz.

- M. Sanchez, J. Aguilar, E. Exposito, "Integración SOA-MAS en Ambientes Inteligentes". DYNA (Q3). Vol. 85, No, 206, pp. 268-282. 2018. https://goo.gl/tEQRgN

- M. Sanchez, E. Exposito, and J. Aguilar. Implementing self-* autonomic properties in self-coordinated manufacturing processes for the Industry 4.0 context. Computers in Industry (Q1), Accepted for publication. 2020. https://i40.page.link/29hQ

- M. Sanchez, E. Exposito, and J. Aguilar. Industry 4.0: Survey from an Integration Perspective. International Journal of Computer Integrated Manufacturing (Q1). Accepted for publication. 2020

Journal (Submitted, in review)

- M. Sanchez, E. Exposito, and J. Aguilar. Autonomic Cycles of Everything Mining for Coordination in the Context of Industry 4.0. Submitted to publication, International Journal of Industrial Information Integration (Q1). In Review. 2020.

Contents

List of Figures

List of Tables

List of Abbreviations

3C	The three last levels of the integration stack (coordination, cooperation, and collaboration)
5C	Five integration levels (connection, communication, coordination, cooperation, and collaboration)
ACL	Agent Communication Language
AI	Artificial Intelligence
API	Application Program Interface
AR	Augmented Reality
BDW	Big Data Warehouse
BPaaS	Business Processes as a Service
BPD	Business Process Diagram
BPEL	Business Process Execution Language
BW	Bandwidth
CDB	Customer Database
CESP	Community Energy System Planning
CMM	Capability Maturity Model
CNN	Convolutional Neural Network
CoT	Cloud of Things
CPI	Machine-Cyber Interface
CPPS	Cyber-Physical Production System
CPS	Cyber-Physical Systems
CPU	Central Process Unit
DBaaS	Database as a Service
DF	Directory Facilitator
DRPM	Resources Provisioning and Monitoring
Eds	Edge Devices
ERP	Enterprise Resource Planning

ESB	Enterprise Services Bus
ETL	Extract, Transform, Load
FIPA	Foundation for Intelligent Physical Agents
HMD	Head-Mounted Display
HMI	Human-Machine Interaction
HSs	Smart Home Servers
IaaS	Infrastructure as a service
IBM	International Business Machines
ICN	Information-Centric Networking
IEEE	Institute of Electrical and Electronics Engineers
IIoT	Industrial Internet of Things
IoE	Internet of Everything
IoS	Internet of Services
IoT	Internet of Things
IP	Internet Protocol
IPCASCI	Intelligent business Processes Composition based on multi-Agent systems, Semantics and Cloud Integration
IT	Information Technology
KaaS	Knowledge as a Service
MAPE	Monitor, Analysis, Plan and Execution
MAS	Multi-agent System
MDE	Model-Driven engineering
MES	Manufacturing Execution System
MP	Mathematical Programming
MV	Minable View
NDN	Named Data Networking
NIST	National Institute of Standards and Technology
OPC	Optimal Program Control
PaaS	Platform as a service
PLC	Programmable Logic Circuits
RAM	Random Access Memory

RCS Resilience Control System

RDF Resource Description Framework

RFID Radio-frequency Identification

SaaS Software as a service

SCEP Supervisor, customer, environment, and producer

SCM Software Configuration Manager

SDC Dynamic Structure Control

SHS Semantic Home Server

SLA Service-Level Agreement

SOA Software Oriented Architecture

SOAP Simple Object Access Protocol

SOM Service-Oriented Middleware

SoMAS Service-oriented MultiAgent System

SPARQL Simple Protocol and Rdf Query Language

SPP Strategic Production Planning

SQL Standardized Query Language

TCP Transmission Control Protocol

UDDI Universal Discovery Description and Integration

VM Virtual Machine

VO Virtual Object

VR Virtual Reality

WSA Web Service Agent

WSDL Web Service Description Language

WSOA Web Service Oriented Agent

XaaS Everything as a Service

Chapter 1

Introduction

Contents

1.1 Introduction

The industry is in the midst of a digital transformation accelerated exponentially by technologies in full growth, such as automation, communication, information, and Artificial Intelligence (AI) (Gökalp et al., 2017; Khan et al., 2017; X. Li et al., 2017a; Suri et al., 2017). These technologies have opened the gates to new ways of production, in which machines, people, processes, and data (the manufacturing processes' actors) involved in manufacturing processes, interact autonomously making intensive use of the Internet and the latest manufacturing technologies, with the primary purpose of developing smarter and more efficient factories (X. Li et al., 2017a; Santos et al., 2017). This new way of production has been called *Industry 4.0,* or *the fourth industrial revolution,* and is considered as a milestone in industrial development that could mark critical social changes in the coming years (Hofmann & Rüsch, 2017; Liao et al., 2017).

The term Industry 4.0 was introduced in the Hanover Trade Fair in 2011 as a concept to denote the new generation of connected, robotics, and intelligent factories. Moreover, Industry 4.0 enables to reconfigure factories in shorter cycles than the traditional ones, making efficient use of human and physical resources (Suri et al., 2017), while increasing the effective communication between the actors involved in the production process (X. Li et al., 2017a). The design principles on which companies and

organizations are based to implement Industry 4.0 are the following:

- Autonomous decisions. The manufacturing processes' actors must be able to make decisions for themselves and perform their tasks autonomously.
- Interoperability. The manufacturing processes' actors have to be able to *connect* & *communicate* with each other and to promote most advances processes for *coordination*, *cooperation*, and *collaboration*.
- Technical assistance. No human actors of Industry 4.0 must offer support and *collaborate* with humans to help them to safely accomplish tasks that can represent a risk for their security, produce fatigue, or are unpleasant. In the same sense, they have to add intelligible visual information to the human so that it can solve the problems in the shortest possible time.
- Information transparency. Information systems must be able to create a virtual copy of the physical world around them using the data collected through their sensors and other devices connected in their ecosystem.

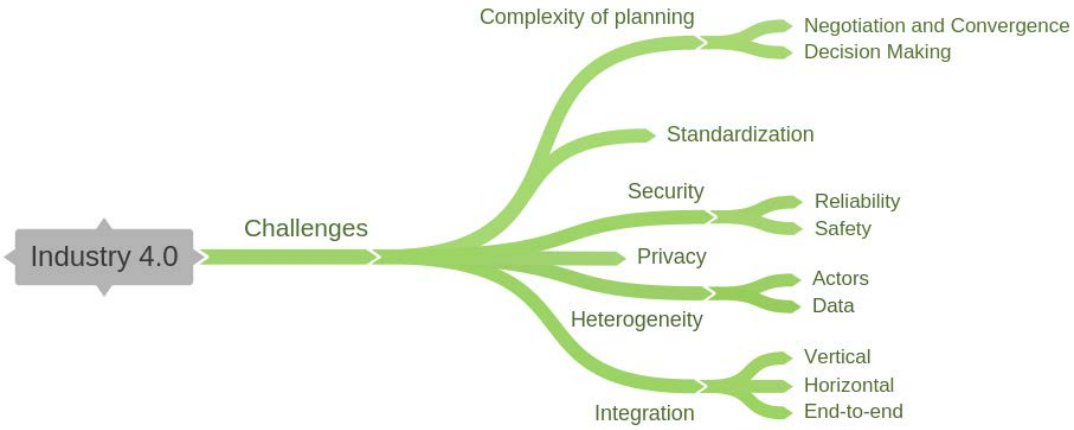

Figure 1.1. Common challenges in Industry 4.0.

Fundamentally, as industry 4.0 is an emerging concept, there are a multitude of challenges, risks and barriers limiting its implementation that need to be solved (Hofmann & Rüsch, 2017; X. Li et al., 2017a; Liao et al., 2017; Lu, 2017; Preuveneers & Ilie-Zudor, 2017; Schwab, 2016; Suri et al., 2017). In that sense, in Figure 1.1, it can be seen the most typical challenges around Industry 4.0. Such is the case of the complexity of planning, for which it must be created explanatory models for managing complex products and production systems (Liao et al., 2017). Another challenge related to Industry 4.0 corresponds to standardization, which means developing common standards to support processes of connection, communication, coordination, cooperation, and collaboration, and in consequence, an architecture to support these standards is necessary (Liao et al., 2017). Another common challenge is related to privacy and security of data, due that sensors and smart devices are continuously collecting information from the environment, and we need to protect that information and avoid unauthorized users to access that data, or still worse, to gain access to control the system (X. Li et al., 2017b; Liao et al., 2017; Preuveneers & Ilie-Zudor, 2017).

Additionally, the heterogeneity of data and actors is considered a big challenge in Industry 4.0 (Liao et al., 2017; Lu, 2017), due that devices from different manufacturers can generate data in a diversity of formats, and might communicate using protocols that are incompatible with other brands.

Notably, this thesis project is focused on solving the Industry 4.0 challenges from an integration and interoperability perspective. Integration is related to link together system components or sub-systems, to allow them to act as a whole and unique system (Drăgan et al., 2017). On the other hand, interoperability is the ability of two systems (or system components) to understand each other and to use functionalities of one into another (Lu, 2017), in such a way that they can work together to produce useful results adjusted to the integration goals. In other words, interoperability allows exchanging information between devices, business processes, interfaces, people, among others (Khan et al., 2017; Lu, 2017; Santos et al., 2017), in order to solve conflicts and achieve agreements in the execution of their tasks. Fundamentally, integrability and interoperability requires from the entities involved to be able to connect (to join each other), to communicate (to exchange information between each other), to coordinate (to follow the orders of a central entity in order to achieve a global goal), to cooperate (to work with others to achieve individual goals) and to collaborate (to work with others to achieve common goals). Another significant challenge, related to integrability and interoperability issues, is how to provide autonomy to the production process, in order to accomplish global production goals. That means that the entities involved in the production processes should be able to promote autonomous coordination, cooperation, or collaboration processes (and some time, to discover processes that have not been explicitly specified).

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). In this context, a central challenge is how to discover the coordination, cooperation, and collaboration necessities, and how to create a management plan to deploy these processes, to allow autonomous integration in manufacturing processes by endowing self-* autonomic capabilities to the system.

Consequently, these integration & interoperability challenges have pushed researches in the last past years. Such is the case of (Jirkovský et al., 2017; Molano et al., 2017), which provided solutions to solve integration challenges at the connection level. They describe how to connect the actors of Industry 4.0 based on the interchange of data. Moreover, Molano et al. (2017) provide a meta-model for integration, allowing

to connect devices through different networks, including social networks. On the other hand, (Jirkovský et al., 2017) uses Semantic Web technologies for data integration on Cyber-physical Systems (CPS), mixed with a Big Data approach to facilitate its implementation. Others relevant researches at this level allow the connection of entities using techniques like Big Data (Khan et al., 2017), AR (Pierdicca et al., 2017; Syberfeldt et al., 2017), or by modifying some network protocols like TCP/IP (Transmission Control Protocol/Internet Protocol) (Bohuslava et al., 2017).

Likewise, at the communication level, Santos et al. (Santos et al., 2017) use a big data analytics approach in order to make the data available to entities, allowing them to communicate indirectly through this data. Similarly, (Suri et al., 2017) provides a human-machine interaction interface allowing entities to communicate using high-level tools, like the Model-Driven Engineering (MDE), which people can use in order to send commands to devices. On the other hand, (Bohuslava et al., 2017) enables communication of robots with a central unit by incorporating existing protocols like TCP/IP into the robot cells. Additionally, Lee et al. (D. Lee et al., 2017) describe how to implement a CPS and how to allow communication and information sharing among devices. Other works at this level are oriented to provide explicit or implicit communications of entities, using a variety of techniques like Multi-Agent Systems (MAS) (Romero et al., 2017), Industrial IoT (IIoT) (Molano et al., 2017; Wan et al., 2016), Big Data (Jirkovský et al., 2017), human-robot interaction (Huber & Weiss, 2017; Nelles et al., 2016), Augmented Reality (AR) (Longo et al., 2017; Pierdicca et al., 2017), etc.

At the coordination level, (Ivanov et al., 2016) introduce a dynamic model to coordinate activities in Cyber-physical supply chains based on the smart manufacturing concept, where a supply chain is modeled mathematically as a networked controlled system. Similarly, (Bohuslava et al., 2017) allow coordinating the activities of robotic cells based on exchanging messages in the form of TCP/IP sockets. Subsequently, Pierdicca et al. (Pierdicca et al., 2017) have developed an android application based on AR in order to allow coordination between peoples and devices. The main goal is to assist one operator in assembling a complex object by using an AR device.

At the cooperation level, Huang et al. (Huang et al., 2017) show how the MAS paradigm can be used to make decisions autonomously, in order to achieve a particular grade of efficiency in a cooperative way, where agents build their self-decision system. It allows agents to negotiate and to improve the energy consumption in a community. Subsequently, (D. Li et al., 2017) is focused on developing a MAS that can deal with the complexity of a cooperative process in smart manufacturing, using a smart factory structure that consists of intelligent agents in which smart objects drive the cooperative process.

Finally, at the collaboration level, Romero et al. (2017) describe how a MAS can be used as a solution to implement a collaboration process autonomously, allowing to perform collaborative tasks in smart factories. In the same way, Riel & Flatscher (2017) proposes a strategic manual method for collaboration among industries, which might

be automatized to allow collaboration processes to be managed autonomously. Furthermore, Richert et al. (2016) make an empirical study of the collaborative problem in human-robot-teams in order to analyze how the appearance of robots influences teamwork. Notably, this research is interesting because they measure the impact of mixing robots and people collaboratively, and they propose recommendations about how to configure this kind of human-robot team properly. Also, Terán et al. (2017) proposes an approach for the integration of industrial automation devices based on MAS, in which the different production units are modeled by agents that interact through protocols that follow cooperation mechanisms. These mechanisms are optimized by using social algorithms. The proposed scheme enables data and service-oriented integration.

The problem statement around the research works described previously will be presented in sections 1.3, 1.4, and 1.5. However, we can say that past research works do not provide a global solution for the integration of all actors involved in manufacturing processes. Mostly, those research works deal with the integration & interoperability of one kind of actor (devices), but, in manufacturing processes, there exists a significant amount of heterogeneous actors as described previously (devices, people, processes, and data). Also, past research works do not propose a well-adapted support for new processes for coordination, cooperation, and collaboration of processes, in a flexible and scalable way.

In this sense, this thesis project proposes a approach to solve integration & interoperability challenges based on the levels of connection, communication, coordination, cooperation, and collaboration, which we have named the 5C integration stack levels. The 5C stack levels will allow solving the challenges around Industry 4.0 incrementally, such that we can group related issues and solve them at the level to which they correspond. For instance, issues related to how to connect heterogeneous actors should be studied at the connection level. However, issues related to the autonomous interoperability of actors might be studied at the coordination, cooperation, and collaboration levels, depending on the actors' interaction and in the manufacturing process needs.

To summarize, this project's objective is focused on solving integrability & interoperability challenges since the incorporation of the autonomous process for connection, communication, coordination cooperation, and collaboration of actors in Industry 4.0. Along with this thesis project, we will study all the elements needed to achieve this objective. In the next subsections, we describe in deep the problematic treated in this thesis project, as well as the proposed solutions.

1.2 Context

The context of this thesis project covers the technologies, concepts, and elements typically used to bring solutions in Industry 4.0 (see Figure 1.2).

- The Internet of Things (IoT). It is a concept that refers to the connections between physical objects, such as sensors or machines, and the Internet (Riggins & Keskin, 2017; Sengupta et al., 2017).

- Industrial Internet of Things (IIoT). It is a concept that refers to connections of devices, the Internet, and their relationship with manufacturing (Wan et al., 2016).

- Data Analytics. Any data analysis technique that could be used to get information from the system or help in the decision making process (Aguilar et al., 2017d).

- Big Data. Refers to large structured or unstructured data sets that can be collected, stored, organized, and analyzed to reveal patterns, trends, associations, and opportunities (Santos et al., 2017).

- Machine learning. It refers to the ability of cyber components to learn and improve on their own through artificial intelligence techniques, without being expressly instructed or programmed to do so (Soto et al., 2019).

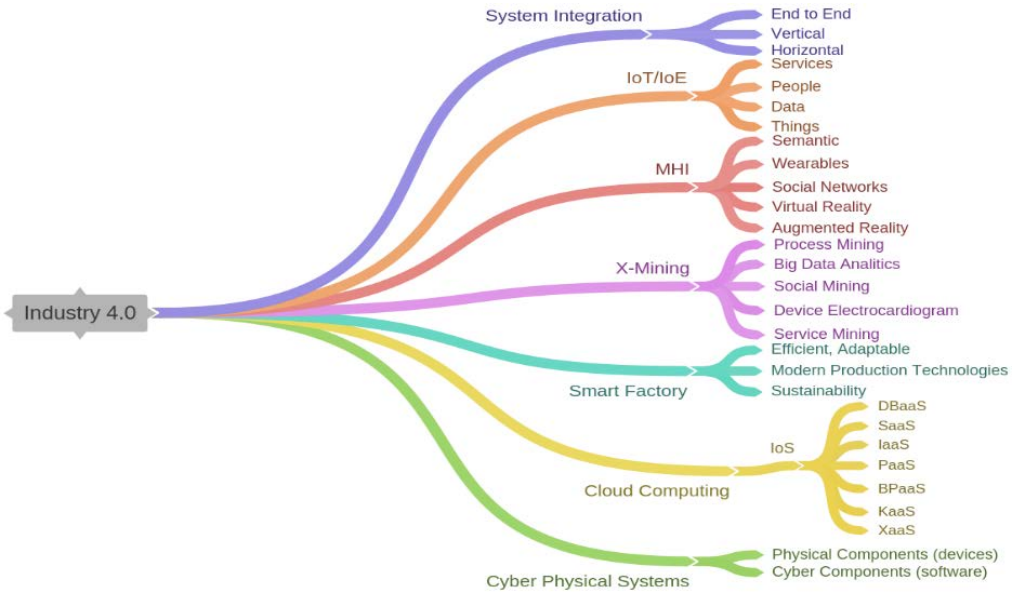

Figure 1.2. Typical technologies, concepts, and elements around Industry 4.0.

- Artificial intelligence (AI). Refers to the ability of cyber components to perform tasks and make decisions that, historically, would require some level of human intelligence. This technology is shortly related to data analysis technology.

- Human-Machine Interaction (HMI). Refers to interfaces allowing humans to take part in the system, but removing the risk that some operations could represent for their life. Furthermore, this technology can also be used to improve the information available to users within a video annotated interface.

- Smart factory. It is a concept commonly used in Industry 4.0. A Smart Factory invests and benefits from the technology, solutions, and approaches of Industry 4.0 (Liu et al., 2017). Smart Factories are developed into intelligent environments in which the real and digital worlds are fully-interconnected (Weyer et al., 2015).

- Cloud computing. It consists of the virtualization as a service paradigm, making a virtual image of somewhat, such as a server, operating system, storage devices, or network resources (Malhotra et al., 2014). Cloud computing is based on three service models that are SaaS (software as a service), PaaS (platform as a service), and IaaS (infrastructure as a service) (Shila et al., 2017; Vaquero et al., 2008).

- Cyber-physical systems (CPS): This is a concept that consists of characterizing each physical object with a cyber component so that this last can act as the smart part, make decisions, interoperate, and execute actions in the representation of the physical object associated with it (Goossens & Richard, 2017).

In this context, Cisco identified different actors that interact in manufacturing processes (X. Chen et al., 2017; D. Lee et al., 2017; Martino et al., 2018; Shaikh et al., 2017; Yang et al., 2017), known as:

- People: It represents humans performing tasks related to the manufacturing process.

- Data: It is raw data produced by other actors in the manufacturing process or historical data stored in organizational data sources.

- Things: It corresponds to devices deployed along the production line, such as machines, sensors, etc.

- Processes: It represents the production process itself, as well as other internal or external processes needed by the manufacturing plan to reach the production goals.

As can be seen, the actors of Industry 4.0 are diverse. However, most solutions regarding the Industry context consider only devices (IoT) as the center of their researches. On the other hand, the technologies around Industry 4.0 are varied. In consequence, the adoption of those technologies within the industry represents a hard task, and if it is not incorporated correctly can lead the automation process to a technological calamity. Particularly, the heterogeneity of actors means that they produce data and communicate in several ways and formats (X. Li et al., 2017a; Molano et al., 2017). It means that if the data is not appropriately treated, it can result in an incorrect understanding of the information shared among actors. Moreover, actors have different needs and goals (individual and collective) for inter-system integration & interoperability that must be fulfilled, in order to manufacture goods or offers

services efficiently. In that sense, autonomous processes for connection, communication, coordination cooperation, and collaboration might help to transform legacy factories into smart ones; however, in the literature, there is not a precise method on how to proceed in that sense.

Therefore, integration among different enterprises is required to create networks of collaboration. Nevertheless, previous researches do not clear on how to proceed in that sense. Furthermore, another vital aspect to consider in Industry 4.0 is that actors must have a high level of autonomy in order to independently act and make decisions. It means that they must be dotted with mechanisms that ensure the self-planning, self-manage, self-supervising, self-healing, etc. of the environment (Huber & Weiss, 2017; X. Li et al., 2017a). Consequently, the actors must have a complete knowledge of the organization and its processes, to be able to make decisions and execute their tasks as required by the Industry 4.0 concept. It means that factories must incorporate mechanisms that allow collecting and creating those knowledge bases from the system (Huber & Weiss, 2017; X. Li et al., 2017a; Molano et al., 2017).

The facts described previously pose diverse challenges that must be considered to develop a solution in the Industry 4.0 context. The next section discusses those challenges in detail.

1.3 Challenges

Industry 4.0 is a concept still in development, so there are many challenges to solve around it. In the same sense, the 5C stack level is an approach proposed in this thesis project that must be used to solve the challenges concerning Industry 4.0 from an integration & interoperability perspective. In consequence, this section presents the Industry 4.0 challenges grouped according to the integration level on which it must be solved, so that we can incrementally deal with them.

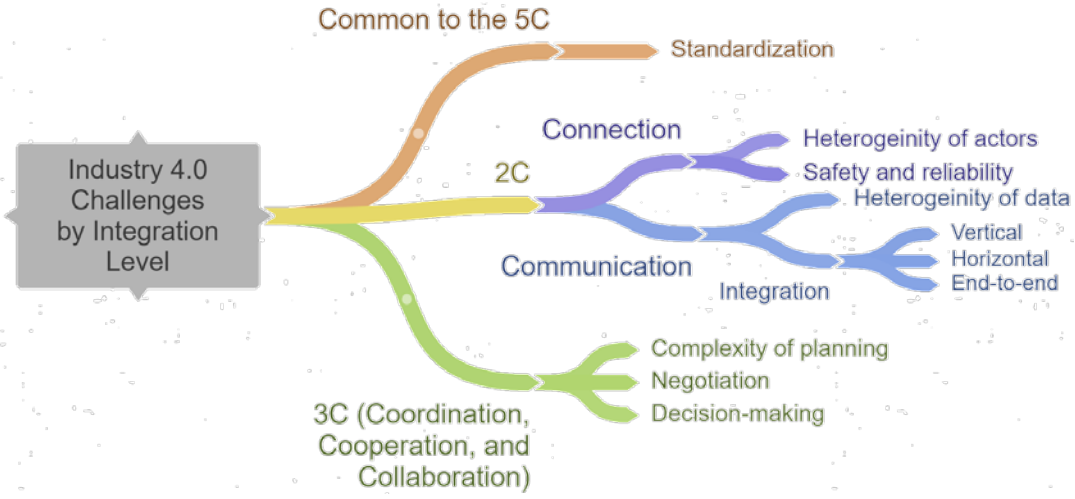

Figure 1.3. Integration Challenges in Industry 4.0 by integration level

Fundamentally, Figure 1.3 shows the main current challenges of Industry 4.0 and

their relation with the 5C stack. It can be seen that the standardization challenge is presented at all levels. Also, in the last three levels, we found challenges more related to planning, decision making, negotiation, and other challenges oriented to allow actors to interoperate. It means to autonomously coordinate, cooperate, and collaborate. Those challenges are detailed as follows:

- The standardization throughout the 5C integration levels. It means that standardized protocols, data & format, etc. are essential to allow a good understanding of the information shared by the actors, to allow autonomous 5C processes to work correctly.

- The definition of different integration styles. Horizontal (between companies), vertical (intra-company), end-to-end (mixing digital and real-world), etc. (Huber & Weiss, 2017; Terán et al., 2017; Yang et al., 2017). Mainly, horizontal integration is needed in order to allow collaboration or cooperation among enterprises. Vertical integration is also required to allow coordination, cooperation, and collaboration between actors involved in the production processes, and the End-to-end integration is required to allow people to participate in the production process in a more natural and less invasive way. It also means coordination, cooperation, and collaboration among people and other actors.

- The specification of negotiation and convergence mechanisms is useful with the intention of allowing actors to ***achieve agreements in the execution of their tasks.*** Notably, some works are proposing autonomic negotiation mechanisms for the 5C highest levels of integration (coordination, cooperation, and collaboration), to autonomously integrate the diverse actors that take place in the industry 4.0, and especially, data, people and services (see Figure 1.3 and Figure 1.4).

- The definition of security and privacy of data mechanisms to offer a robust integration of actors is essential to provide some mechanisms that preserve the security and the privacy of the data, in order to avoid an unwanted attacker to gain access to the information of the production process (Huber & Weiss, 2017; Jirkovský et al., 2017; Molano et al., 2017).

- The management of the heterogeneity of actors (connection) and data (communication) is essential due that different actors generate a large amount of data, information, and knowledge in different formats, which should be semantically integrated to allow actors to understand the messages correctly (X. Li et al., 2017a; Molano et al., 2017).

- The management of the complexity of planning implies that smart factories will allow the self-configuration of the manufacturing process (Huber & Weiss, 2017; X. Li et al., 2017a).

- The management of the decision-making process is oriented to promote the independence of actors by adding the ability to make autonomous and optimal decisions related to executing the tasks that better help to reach their goals (individual or collective) (Molano et al., 2017).

The next section presents a start-of-the-art focused on showing how past researches are dealing with challenges in Industry 4.0 at each integration level.

1.4 Related works

In this section, a selection of articles related to Industry 4.0 is presented. This state-of-the-art is organized according to the actors/dimensions of the IoE (people, data, things, services) versus the 5C integration stack levels. This start of the art will help to identify which integration levels and actors were the most studied. Moreover, this state of the art will help to recognize the fields of study that need more attention.

Mainly, we have presented the 5C stack levels as a method to solve the challenges of Industry 4.0 incrementally, from an integration perspective. The levels of the 5C stack are presented below:

- Connection: it allows entities of IoE to be linked to the network and to share a standard media or a channel for communication. That means that it allows the actors to contact each other. The connection is essential to allow communication.

- Communication: Once entities are connected, they can exchange messages, allowing them to establish a conversation and interactions. Also, communication means that the entities can understand each other. Connection and Communication are essentials to achieve interoperability of the system, as well as to allow the coordination, cooperation, and collaboration processes.

- Coordination: Basically, the coordination is an activity carried out by a central entity (or orchestrator) that allows coherently to harmonizing the execution of the tasks of a system (intra-systems integration or vertical integration). In terms of services, the coordination is closely related to the concept of intra-system orchestration (internal to a business process or system).

- Cooperation: It consists of a negotiation process that allows the entities of the same system (intra-system integration or vertical integration) or entities of two or more systems (inter-systems integration or horizontal integration) to achieve agreements in the execution of their tasks, in order to accomplish individual objectives. Cooperation is related to inter-system choreography.

- Collaboration: refers to entities 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 working alone. Collaboration is related to extra-system/inter-system choreography (interactions between autonomous processes). Collaboration does not rely on a central unit.

According to this classification, a survey of research works mainly focused on the levels of integration of the IoE actors, was elaborated. In that sense, Figure 1.4 shows a summary of the recent researches classified by the integration level. In this case, a blue bar indicates the number of research works dealing with the integration of Things. An orange bar means the number of research works that focus on the integration of Data. The gray bar specifies the number of researches that considered the integration of people, and a yellow bar corresponds to the number of papers dealing with the integration of services. For instance, at the connection level, three research works can connect things, three connect data, three connect people, and two researches interconnect services. Also, one paper can deal with the integration challenges at many levels, as described below.

Furthermore, at the coordination level, two researches worked with the coordination of Things, one paper considered the coordination of people, and one research dealt with the coordination of services. Also, in the literature review, it was found only one paper dealing with the integration challenges in more than one level. It means that any solution provided by previous researches can allow all actors to connect, communicate, coordinate, cooperate, and collaborate.

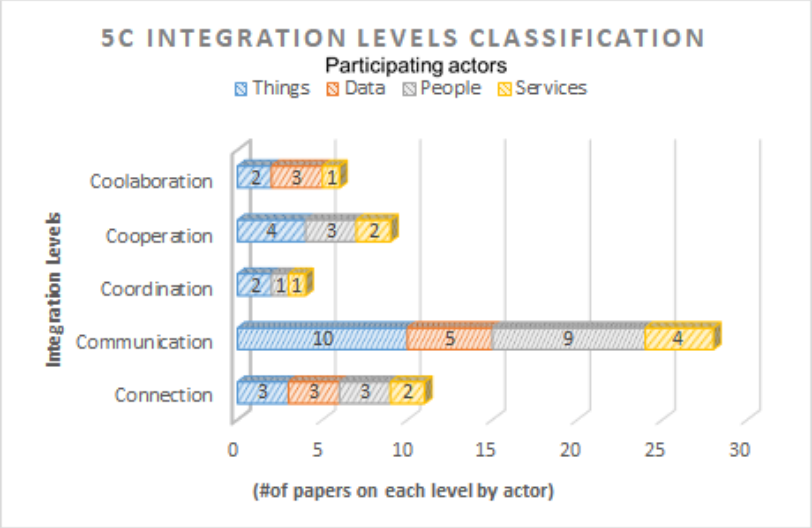

Figure 1.4. Existing works classified by the four actors of IoE vs. the 5C integration levels.

In the next sub-sections, these research works are analyzed according to the 5C level that best suited to each work. The works have been selected because they cover several aspects of the proposed classification model, or they have some relationship with the problem of autonomous integration in Industry 4.0, mainly for the highest 5C levels (coordination, cooperation, and collaboration). In this classification, it can be noticed which integration level has been the most studied in recent years, as well as the actors involved at each integration level.

1.4.2 Connection.

Research works positioned at this level focus mainly on the interconnection of entities, using a shared medium, but without defining explicit mechanisms for communication, coordination, cooperation, or collaboration and without solving integrability requirements.

Molano et al. (2017) describe an architecture for IoT applied to the industry (denoted as IIoT, for Industrial Internet of Thing), which integrates IoT, sensors, actuators, social networks, and cloud computing. The prototype architecture contains five layers. The Sensing layer comprises several types of devices, and it is responsible for collecting data from sensors or other devices, as well as to manage the manufacturing and logistics processes. The Database layer contains the physical (SQL and NoSQL databases) and virtual databases (logical links to the databases in the network nodes). The Network layer supports all the infrastructure components (physical devices), allowing devices to connect using wireless or wired networks.

Moreover, the network layer allows sharing the data with other devices connected to the system, enabling the interaction between the Sensor layer and the User layer. Similarly, the Data Response layer represents a data set whose goal is to keep the persistence of other layers. The User layer contains the Application Program Interface (API) used by ERP applications, in order to monitor the raw material, the equipment failures, the quality control and programming of the production. This layer is a Middleware that provides several services, such as data compilation, transmission, data processing, IoT services, etc. Molano et al. (2017) focused on the interconnection of data with things through services, which access databases in cloud computing and make data available to each device. It means that devices are connected through the data. However, this work does not include the coordination topic between the integrated actors. Because of that, this thesis project considers it to be positioned between the connection and communication levels.

On the other hand, Jirkovský, Obitko, and Mařík (2017) uses Semantic Web technologies mixed with a Big Data approach for data integration for CPS. They use a Semantic Web approach to deal with the semantic and platform heterogeneity issues. The authors propose an ontology (called SHS ontology) for the description of the industrial data. The SHS ontology contains structures that allow modeling different observations: the physical qualities, the units of measurements, or the external data sources. The architecture is divided into four main parts. First, the *Data acquisition layer* collects data from sensors, other systems (Manufacturing Execution Systems (MES) or Enterprise Resource Planning (ERP) systems), and some relevant external data sources. Besides, this layer is in charge of solving platform heterogeneity issues. Next, the *Transformation layer* converts data to a unified semantic form, according to the SHS ontology (it fixes the damaged data if needed).

Moreover, in the *Transformation layer*, the semantic heterogeneity is solved, and Resource Description Framework (RDF) triples are created, in order to populate the

data with semantic information. The *Data storage layer* is in charge of storing the triples in the RDF format. Finally, the *Analytic layer* provides direct access to the storage layer for analysis tasks or customizing the user queries. The authors affirm that the main advantages of the integration using the SHS ontology is that the ontology describes the reality in its representation, and data can be easily queried in SPARQL. Same as previous works, this research is focused on put data available to other actors and dealing with the data heterogeneity issues. They allow the connection between actors. For that reason, this thesis project classifies it as belonging to the connection level.

Notably, (Jirkovský et al., 2017; Molano et al., 2017) are relevant for our research, because they describe how to connect actors of the industry 4.0 through data. Other works in this domain allow the inter-connection of actors using techniques like Big Data (Khan et al., 2017), AR (Pierdicca et al., 2017; Syberfeldt et al., 2017), Network protocols like TCP/IP (Bohuslava et al., 2017; Exposito, 2013), or works that present architectures for CPS integration (J. Lee et al., 2015).

1.4.3 Communication.

Most of the studied works are classified at this level because they allow entities to communicate, making abstraction of underling connection details, and without proposing explicit processes for coordination, cooperation, or collaboration.

Román et al. (2018) proposes a communication layer aimed to retrieve data of robotic arms manufactured by different firms. The proposed system allows monitoring the status of robotics cell in a footwear factory and displays a 3D visor that shows a simulation of the movements of robotic arms. Essentially, this research work unifies the different communication protocols used by manufacturers into a single one. Moreover, they have defined a custom communication protocol over TCP/IP to retrieve the data from the monitoring system. This protocol supports 128 types of messages, but only five have been defined (MSG_TIME, MSG_NOTIFY, MSG_DOF, MSG_JOINTS, and MSG_SELECT). MSG_TIME is used to maintain the timeline of the monitored data. MSG_NOTIFY allows sending notification messages from servers to clients. MSG_DOF is used to change the number of degrees of freedom of robot arms' joints. MSG_JOINTS allows maintaining the angle value of each joint in the robotic arm chain. Finally, MSG_SELECT allows a client to collect data from a specific robotic arm. The communication messages are protected from unwanted attackers. Thus, all the data shared between servers and clients is encrypted to avoid man-in-the-middle attacks. This research work corresponds to the communication layer, due to that its goal is to allow robotic arms from different manufacturers to communicate using a standard protocol.

Santos et al. (2017) propose a Big Data Analytics architecture that includes layers dedicated to deal with data needs, from the collection to the analysis and distribution. The proposed architecture is divided into seven layers. Components define each layer, and each component can be associated with some technological tool. The *first layer* represents the Big Data producers and consumers' entities; these entities are usually

consumers of raw data, indicators, or metrics. The *second layer* (Data sources layer) represents the different sources of data, including components such as Databases (operational/transactional databases), files, ERPs, E-Mail, sensors, among others. All this data will feed the ETL layer (extraction, transformation, and loading processes). The *third layer* corresponds to the process of extracting data from data sources and storing it into the Big Data Warehouse (BDW), using several technologies to implement the ETL process and to integrate data from multiple data sources. The *fourth layer* is the Data Storage layer. This layer was divided into two sub-layers, which contain different components that must be used according to the context. Consequently, the data storage sub-layer is in charge of storing data into a NoSQL database, like Cassandra or HBase streams in real-time.

On the other hand, The Hadoop BDW sub-layer is in charge of preserving the historical data. Once the data was stored in the BDW, it will be available for data analytics through the SQL query engine. The *fifth layer*, the Raw Data Publisher, enables access to the data by providing Web Services for the data stored in the Data Storage layer. The *sixth layer*, the Big Data Analytics, includes components to facilitate the analysis of the data, making available different data analysis techniques like Data Visualization, Data Mining, Reporting, etc. The *seventh layer* applies a mechanism for Security, Administration, and Monitoring.

Moreover, this layer includes components needed in the other layers in order to guarantee the proper operation of the whole infrastructure. In general, this work allows the data to be available to other actors, letting them communicate indirectly through it. However, this research does not study coordination processes, nor proposes processes for cooperation and collaboration directly. For this reason, it is considered to be positioned at the communication level. Also, it considers only the data dimension of IoE. Moreover, even if the author claims this works are adapted to Industry 4.0 (Santos et al., 2017), they did not present a precise application on this domain.

Similarly, Suri et al. (2017) provide a solution for the modularity and interoperability issues related to Industry 4.0 from a systems integration viewpoint, focusing on the "vertical integration" of system using the model-driven engineering (MDE) approach. This approach enables heterogeneous systems to communicate in a low-coupled manner. In particular, this approach is oriented to industrial robots, which perform standard repetitive tasks. The communication model consists of two layers. The *model-based behavior layer*, in which the task execution model is created using an activity diagram in UML 2.0; and the *robot's implementation layer*, which is in charge of transforming the activity diagram designed in the previous layer into instructions recognized for robots by using some available frameworks designed for this purpose (i.e. Papyrus). The execution of the robots is made using API calls through the execution model (on-line execution), rather than deploying the source code on the system (off-line execution). In this sense, this approach allows the creation of complex systems of sensors and actuators, with low computational power and low energy usage.

Consequently, the main objective of the previous research is to allow things to communicate transparently and in a loose-coupled way. Due to this reason, it has been classified in the communication level. This research work is relevant for our research because it shows how to integrate and communicate actors of the industry 4.0 using an MDE based approach, to send orders to devices.

On the other hand, Bohuslava, Martin, and Igor (2017) enable communication based on the standard Ethernet (IEEE 802.3) for the control of the robotic cell. The communication protocol used in this model of production robot cells is TCP/IP. The Control application is divided into several parts, consisting of a server running the control task, and some client subprograms running on each robot. All the requirements of the cells, as well as the instructions from the control center, will be processed as TCP/IP sockets. Complete communication with the sockets takes place only through the central unit, which coordinates the communication among the robot cells. This coordination is based on the messages exchanged thought the sockets, according to two variants: a) Confirmed coordination, where the completion of each operation is notified to the central unit b) Unconfirmed coordination, in this case, the continuity of the activity is not conditioned to receive a confirmation message from a superior object. In this sense, robot cells can communicate using the TCP/IP protocol, with a central unit making the coordination of the whole communication process. This research work deals with problems of connection, communication, and coordination of robotic cells; however, the authors of this research work said that the primary goal of it is to allow those robots to communicate. It is the reason why this thesis project considers it to be in the communication level. This research work is significant for our research because it describes how to incorporate an existing low-level communication protocol in new devices, to allow them to communicate appropriately.

Additionally, Lee et al. (2015) propose a unified 5-level architecture as a guideline for the implementation of CPS. The proposed 5-level architecture provides a step-by-step guide for developing and deploying a CPS in manufacturing environments. The first level, the Smart connection level is in charge of acquiring the data directly from the sensors, or of collecting it from the controller or the enterprise manufacturing systems, such as ERP, MES, SCM (Software Configuration Management) and CMM (Capability Maturity Model). The second level, the Data-to-information conversion level, infers useful information from the data using several tools and methodologies. The Cyber level acts as a central information hub. The information is sent from every connected machine in the network to the Cyber level.

Moreover, specific analytics tasks are used to extract additional information that provides better insights regarding the status of each machine. The cyber level uses a machine–cyber interface (CPI), in order to allow the interconnections between machines. On the other hand, the cognition layer generates detailed knowledge of the monitored system, and makes it available to experts, allowing them to make the correct decisions. According to the authors, this level requires proper user interfaces/dashboards, in order to transfer the acquired knowledge to the users completely. The configuration level allows self-configuration and self-adaptation of

the devices, by getting feedback from cyberspace to physical space, acting as supervisory control. That configuration allows applying the corrective and preventive decisions (defined in the cognitive level) to the monitored system. In that sense, this system acts as a resilience control system (RCS). This research presents an intelligent middleware, which allows the coexistence of devices and people, facilitating the collection and transformation of data between them. Due to that, this research is positioned at the level of communication. Consequently, this research work is relevant for our research because it describes how to deploy a CPS and how to communicate and share information among things.

Other works in this domain allow explicitly or implicitly communications between actors, applying a variety of techniques, like MAS (Romero et al., 2017), IIoT (Molano et al., 2017; Wan et al., 2016), Big Data (Jirkovský et al., 2017), human-robot interaction (Huber & Weiss, 2017; Nelles et al., 2016), AR (Longo et al., 2017; Pierdicca et al., 2017), or Middleware (Ferrera et al., 2017).

1.4.4 Coordination.

At this level, this thesis project considers only those paper that allows coordination process using a central coordinator, according with the requirements defined previously (see Section 1.4), for the coordination level (centralized intra-system coordination). In that sense, Orellana & Torres (2019) propose a procedure to transform a legacy manufacture process into a smart factory level 2, according to Industry 4.0. Essentially, this proposal allows vertical integration, which guarantees the actors involved in the internal production process to share information. Notably, the remarkable point of this proposal is that it grants integration without buying new machinery. Moreover, their proposal uses IIoT to achieve its integration goals. The procedure comprises eight steps that are executed continuously until the industrial process works correctly.

1. Define management indicators. Define the indicators that will be used for evaluating the process. The authors propose to use the ISO 22,400 standard (Kang et al., 2016) for this purpose.

2. Define the processes' inputs, signals, and sensors. This step helps to determine which measurement instruments are required by machines and how to link them with other machinery's sensors.

3. Identify and choose data sources based on the corrective and preventive maintenance. In this step, the data source associated with each indicator is defined.

4. Link equipment of new and old machines. Determine the proper equipment required for each machine being automated in order to allow it to operate autonomously.

5. Create standalone networks. Machinery might communicate using a dedicated and independent network in order to avoid communication conflicts.

6. Generate alerts when processes' variation is detected. An alarm must be generated when the system encounters a fault, such as the shutdown of a motor, actuators, among others.
7. Improve feedback and follow up processes. The production orders in which operators are working, as well as operation's errors and problems, are captured using a data collection software and send to an ERP software, such that all the information about the process can be available when it is requested for failure diagnosis.
8. Test and validate the system. Verify if the system operates correctly or needs to be tuned up.

The case study for Orellana & Torres (2019) was conducted in an enterprise where the machinery had more than 47 years of operation. The results show that after the production process was transformed into a smart factory level 2, the production line was able to reduce the average production time from four days to three hours. This research work is positioned at the coordination level because it improves the coordination of the whole production process.

Soto et al. (2019) present an orchestration framework that combines IoT and machine learning for failure detection in production lines. The solution comprises three fundamental elements. Firstly, the production line is exposed using an IoT Connector that is responsible for transforming the data from different production protocols into network protocols or message queue systems. Secondly, the connector propagates the data using a Broker according to the IoT standards. Thirdly, the data is processed using a learning agent that orchestrates all the components' behavior and selects the learning algorithm depending on the data characteristics and the current use case.

Soto et al. evaluated holistically this framework, using a realistic simulation. However, in a real production line, issues not covered by this solution are the heterogeneity of actors, the bottlenecks, the failures not considered in the machine learning model, etc. This research work is positioned at the coordination level because it can orchestrate the behavior of the production line. However, there is still much work to do in order to promote autonomous interoperability.

Ivanov et al. (2016) introduce the dynamic control concept and a dynamic model to coordinate activities in cyber-physical supply chains, based on the smart manufacturing concept. The authors propose a scheduling approach based on making a temporal decomposition of the scheduling problem to allow the dynamic execution of the jobs. They use a dynamic structure control (SDC) approach model, which is a dynamic interpretation of planning, in concordance with the execution time. Additionally, SDC is combined with the optimal program control (OPC) theory and mathematical programming (MP). The dynamic of planning is because the decisions on supply chain planning are taken for specific intervals of structural constancy. In this sense, a static optimization problem is solved with the help of MP for each time

interval, while OPC is used in order to define and to model the transitions between the time intervals.

Moreover, the supply chain is mathematically modelled as a networked system described through control models M1-M2 (schedule for material supply processes, schedule for services, respectively). Then, when the manufacturing process starts, the M1-M2 process assigns services to business operations in sequential order. Next, M2-M3 (M3: schedule for resources) assigns and schedules services to information resources. Finally, M3-M4 (M4: schedule for information systems modernization) is launched, in order to reconfigure the system. The coordination happens in the system's interconnections; for instance, the output of M1 is used in the constraints of M2; Analogously for M2, M3, and M4. This research is positioned at the coordination level because a central entity executes the coordination process. Also, this research work is essential for our research because it shows how the coordination processes can be autonomously deployed in the industrial domain.

Pierdicca et al. (2017) develop an AR Android application, in the context of the Industry 4.0. This application assists an operator to let him assembling an object composed of several components that must be linked together in a specific order. At the end of the assembly phase, the application makes a verification of some parts of the final object. In this sense, the application displays the assembly instructions, one by one, using the head-mounted display (HMD) that the user wears. Moreover, the application uses textual information and 3D models of the real scene in order to help the operator to finish the task quickly. That means that for implementing this application, some 3D models for each real component must be created; those models must keep the same dimensions and components as the real scene.

Moreover, the 3D model of the real object is created using different colors for each component to allow users to recognize them easily. This research connects people with things in a coordinated way, in which the android app is the coordinator that displays information through the HMD interface to allow users to assemble an object more comfortably. This research work is positioned at the coordination level because it allows achieving a global goal using a central coordinator. Moreover, (Pierdicca et al., 2017) present an unusual approach that combines AR with devices and people to enable coordination, which can be very useful in the context of Industry 4.0.

Another research work with some level of coordination is (Bohuslava et al., 2017); however, that work is focused mainly on the connection of things.

1.4.5 Cooperation.

The papers positioned at this level promote a specific mechanism for cooperation. For example, Huang et al. (2017) propose a community energy system planning (CESP) model based on a Multi-Agent Systems (MAS), in order to improve the energy utilization within a specific community. In the solution, each participant is viewed as an agent. Furthermore, the stakeholders are represented as CESP agents too. Moreover, this solution considers four types of stakeholders into the model:

Governments, People, Energy firms, and Energy facilitators. Additionally, the spatial location is also considered into the model, due to the transmission cost of hot and cold water. In this sense, all agents are organized in a spatial hierarchy and divided into different groups based on whether they have similar interests or not.

The negotiation process is only performed between agents of the same group, in order to improve the negotiation efficiency and to reduce the negotiation time. The information needed for the negotiation process, like price, supply and demand, policies, and other agents' initial planning, etc., is available to all agents. That is, they share their belief-desire-intention data within the group. Mainly, this research allows agents to build a decision-making system, which lets agents performing negotiations in order to improve the energy consumption within a community. Thus, each agent of the platform can find potential partners, negotiate and construct its decision-making model, with the primary goal of making an optimal decision related to the energy consumption within the community. The information needed by the agents to build their decision-making model includes databases, as well as negotiation models (such as persuade, threaten, inducements, and promise). Every agent uses a specific format that the other agents know to uploads the negotiation models. However, in order to guarantee the privacy of the data, private information is only visible to interrelated agents. These agents cooperate without a central coordinator, in order to achieve the objectives discussed previously; because of that, it is classified at the cooperation level. In the same way, this research work shows how enterprises can offer their services as cloud-computing services, and how customers can select the service with the best benefits autonomously. Particularly, this research work is significant for our research because it shows how the MAS paradigm can be used to make autonomous decisions and achieve a specific grade of efficiency cooperatively.

Also, D. Li et al. (2017) focuses on developing a MAS that can deal with the complexity of the cooperative processes in smart manufacturing, using a structure consisting of intelligent agents with cloud computing-based feedback and coordination assistance. The negotiation mechanism allows a smart product (instantiated as an agent) to act as manager, while the smart machines and smart conveyor belts (instantiated as agents too) acts as contractors, competing by tasks. In this case, a Radio-frequency Identification (RFID) tag is used for the communication of the agents, by reading/writing it. Smart machine agents or smart conveyor belt agents initiate the negotiation process. The smart product agent publishes a task, and the smart machine agents receive it, deciding whether to bid for the task or not. The product agent uses a rule-based decision system in order to calculate the performance indicators that allow deciding which contractors are awarded or rejected. The contractor (smart machine) that wins the bid will execute the task. After the smart machine agent is selected, it is necessary to start moving the smart product from the current location to the target location, by constructing a conveyor belt route chain. Again, a new negotiation round is required in order to create the route from the start point to the target point. This research is focused on the cooperative process to organize smart products and smart devices into a smart factory, by using the MAS paradigm where each agent has individual objectives. This research is classified at the

level of cooperation because the process of coordination is not centralized, like was defined in section 1.4. Besides, this work is relevant for our research because it gives us a vision of how smart objects can drive a cooperative process.

Other works in this domain propose cooperation processes using technologies like MAS (Romero et al., 2017), MDE (Suri et al., 2017), or human-robot interactions (Nelles et al., 2016).

1.4.6 Collaboration.

In this sub-section, work focused on the study of explicit collaborative process is presented. That means interactions between system actors in order to achieve a common goal. In that sense, Romero et al. (2017) propose a social factory architecture based on adaptive, collaborative, and intelligent MAS. Moreover, they explore the role of what they name a social operator 4.0 in the context of smart and social factory. Mainly, people, devices, and software systems socialize together (cooperate or collaborate) in real-time, to support manufacturing and services operations. The authors define a social operator 4.0 as a type of operator that combines smart wearable solutions in conjunction with advanced human-machine interaction (HMI) technologies to promote cooperative/collaborative processes with other social operators, social machines, and social software systems. MAS is used to simplify the communication between the cyber-physical elements, which means humans, machines (real world), and software entities, as well as to distribute tasks (based on their competencies) and share & trade control in collaborative tasks.

Moreover, the MAS keeps as much as possible human inclusiveness within the manufacturing process, without compromising production goals and efficiency. Finally, the MAS can improve the skills of the human and machines through learning and practice, for what it must record and track their evolution. The social factory MAS is composed of human agents that characterize humans and their skills; artificial agents (machines) that characterize the machines and their capabilities; interface agents that characterize interaction rules and conditions for assisting humans and machines interfacing with the rest of the system. Similarly, Broker agents characterize the levels of automation available in the system and the rules for sharing and trading control in human-machine cooperation, in order to efficiently allocate and distribute tasks between the cyber part and the humans at the workstations of the manufacturing system. This research provides a suitable mechanism for facilitating collaborative tasks in smart factories, due to this reason, it is positioned at the collaboration level. This work is relevant for our research because it shows how MAS can be an excellent solution to deal autonomously with the collaboration issues.

On the other hand, Riel and Flatscher (2017) deals with the creation of a structured methodological approach to strategic production planning (SPP), intending to establish an integrated manufacturing road-mapping process. The essential idea is that the stakeholders involved in an integrated design process shall run a series of stages of divergent and convergent thinking. At each stage of divergence, the ideas about the design process are generated out-of-the-box thinking. Then, in the convergence stage,

the ideas are consolidated and evaluated, with the central goal of deciding how to proceed with every single idea. Typically, multiple parallel paths are created, where each path represents a particular set of ideas being worked. For the particular design problem of the SPP, the authors proposed a schema with three phases (each phase involves divergence and convergence stages). The first phase is in charge of identifying the most relevant topics, based on the use of techniques like brainwriting, extreme scenarios, etc. In the second phase, the participation of the top management representatives is required to prioritize the topics, according to the company strategy. The third phase uses the ranked topic list as input, to deal with the specific challenges linked to each topic. Next, the experts are involved in the process to identify and group each selected topic focusing on what has to be done rather than on when.

Consequently, the goal of the final phase is to define a concrete plan addressing the requirements and problems detected for each challenge. Finally, this research work is a proposal for a strategic method for collaboration among industries. The authors of (Riel & Flatscher, 2017) have designed a use case to demonstrate how this proposal fits in Industry 4.0. However, this is a manual process that does not involve any kind of automation. This research is motivating for our purposes because it shows a method that could be automatized to allow autonomous collaboration processes in Industry 4.0.

Likewise, Richert et al. (2016) make an empirical study of the collaborative problem in human-robot-teams. The method uses virtual reality to simulate a task that can only be accomplished through teamwork between robots and humans. The participants are immersed in a virtual scenario developed in Oculus Rift (a virtual reality system). The objective of this experiment is to analyze how the appearance of robots affects teamwork outcomes. In that sense, a humanoid robot is used in one experiment and an industrial robot in another. The human forms a team with each robot, and his physical reactions are studied while the team develops a task collaboratively. A set of instructional commands is provided beforehand to the human, in order to allow the communication with the robot.

On the other hand, Oculus Rift collects all the data generated during the study. The authors suggest using big data analytics in order to understand the nature of the collaborative process better. However, they consider that these experiments are only the stepping-stone to much more detailed research. Although the authors of (Richert et al., 2016) say that this research is linked to Industry 4.0, they do not clarify how this method can be used in this context to increase production, reduce costs, among other things. Essentially, our interest in this research work is because they measure the impact of mixing robots and people collaboratively.

The next section will discuss the issues around the Industry 4.0 context regarding this state of the art, to present the proposed solution that must be developed in the next chapters of this thesis project.

1.5 Positioning

The related works presented in Section 1.4 shows how some of the main proposals are addressing the integration challenges in Industry 4.0. Despite the significant number of works in this area, we could not find a complete solution that covers all the integration aspects through the 5C levels in a production process (see Figure 1.4). Furthermore, it is a fact that a process composed of two or more actors instances might be able to connect, communicate, coordinate, cooperate, and collaborate, in order to achieve the production goals efficiently, minimizing conflicts between the actors' task execution (Pietrewicz, 2019). We have summarized these findings in Figure 1.5, which shows that the most studied integration level is communication, followed by connection and cooperation levels, leaving at the last place the collaboration and coordination levels. It means that researches are focusing mostly on the first levels of the 5C stacks, leaving many unfinished work regarding the self-organization of manufacturing processes.

Moreover, as discussed at the beginning of section 1.4, any solution allows integrating and encouraging interoperability processes for all actors, mainly because past researches use IoT as integration technology, which considers Things as the only actor. This fact means that processes of planning, negotiation, decision-making, knowledge extraction, and others, will not have a global vision of the whole production chain because they miss useful insights that could be extracted from actors like people, data, processes, and also services.

Additionally, in previous researches, it is not clear how to extract information from the manufacturing process actors and how to proceed to endows autonomy and proactivity to the whole system.

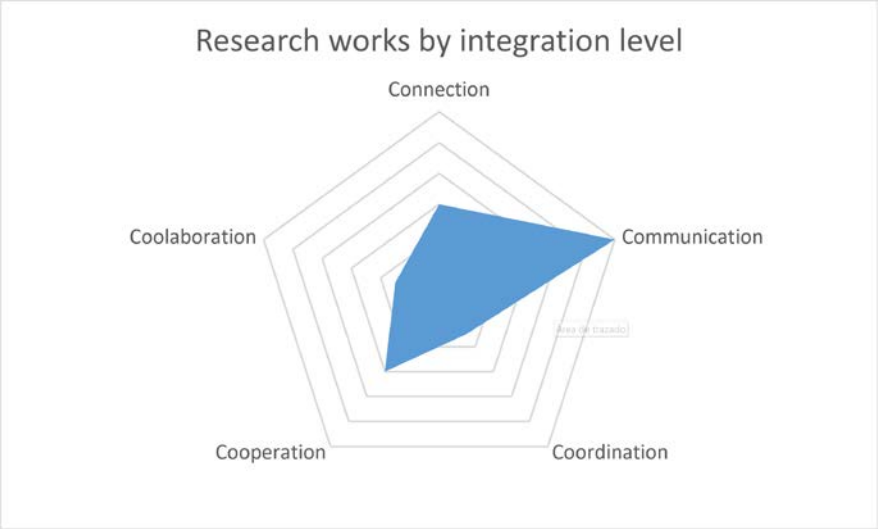

Figure 1.5. The research works by integration level.

In that sense, the main objective of this thesis project is to propose a general framework to solve the integration and interoperability issues in the Industry 4.0 context. This framework will . This idea of autonomous integration of actors in Industry 4.0 will comprise three key elements:

- The Internet of Everything (IoE) paradigm. Notably, the IoE was defined by Cisco as "the Internet that connects people, data, processes, and objects (things) together, giving connectivity of anything-anytime-anyplace for more intelligent health, energy-efficient smart cities, smart transportation, among others" (X. Chen et al., 2017; Martino et al., 2018; Shaikh et al., 2017; Yang et al., 2017). For instance, the IoE must allow us to integrate the employees involved in the production process (people), the production process itself (business process), the devices that perform the production task (things) and the information needed by them (data), in order to perform their tasks properly. Mainly, the IoE must provide solutions to the integration issues at the level of connection and communication, but we must still consider the interoperability issues, in order to allow the self-management of coordination, cooperation, and collaboration processes.

- The autonomic computing paradigm must allow the production process to be self-configured before starting up (ex. creating the coordination, cooperation or collaboration plans), and during the execution be self-optimized (ex. reducing resources consumption), self-healed (ex. fixing failures) or self-protected (ex. by providing privacy and security of data). In general, the "Everything Mining" tasks can extract information and knowledge about the processes, data, people and things, in order to autonomously organize the production process, make smart decisions, among others, to reduce the production costs, and improve the use of raw materials, etc.

- The Everything Mining techniques will be useful as resources to deploy coordination, cooperation, or collaboration processes that take place within the manufacturing processes. The "Everything Mining techniques" is a concept used in this thesis project, which groups any mining technique that can be used to discover useful information, like data mining, process mining, services mining, sentiment analysis, semantic mining, among others. Notably, the X-Mining techniques should help to build the knowledge bases needed by autonomic computing. Consequently, X-Mining is based in an intelligent control loop known as MAPE+K model (Monitoring, Analyzing, Planning, Executing and Knowledge) (Vizcarrondo et al., 2017), in order to be aware of the environment and to endow the production process with mechanisms of self-configuration, self-healing, self-optimization, and self-protection, etc. (J. Lee et al., 2014).

1.6 Contributions

The challenges found in section 1.3 guide the contributions of this thesis project. Those contributions are listed below:

1. One of the main contributions of this thesis project focused on the

communication level of the 5C stacks. It presents an architecture for the integration of Multi-agent systems and the IoS paradigm. Additionally, this architecture is extended with the fog computing paradigm oriented to deal with the requirements of real-time, low latency, and other issues related to any cloud-based system. Besides, this thesis project gives one example of autonomous communication in the context of Industry 4.0 using the presented architecture.

2. The second contribution of this thesis project is a framework for autonomous integration of actors in Industry 4.0, using technologies like the Autonomic Computing, the Internet of Everything, and the X-Mining paradigm. This contribution is ranged at the third higher layers of the 5C stack (coordination, cooperation, and collaboration).

3. Additionally, we present three autonomic cycles of data analytical tasks that provide self-coordination in manufacturing processes. These autonomic cycles intends to provide integration and interoperability of actors in Industry 4.0. It means that data, people, things, and services can autonomously work together to achieve the production goals coordinately. This contribution is ranged at the third level of the 5C stacks (coordination).

4. We present and discuss the applicability of the proposed framework in a traditional industry so that it is possible to transform it into a smart factory.

5. Furthermore, this thesis project presents the implementation of a self-supervising autonomic cycle for manufacturing in a coordination context 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. Other useful information provided by the Process mining corresponds to the bottlenecks presented in the manufacturing 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.

    - A machine learning technique is used to build a predictive model that detects if a product will fail or not the quality control test. This result will allow the system to perform an auto reconfiguration 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.

1.7 Structure of the Thesis

This thesis project is organized following the principles of writing the thesis by articles adopted by the doctoral school on July 3, 2017 (Commission Académique, 2017). In consequence, it is proper to mention that this thesis covers the requirements required by the academic commission to be considered as such. First of all, I am the first author of the five articles produced during this doctoral research. Secondly, the co-authors have given the right to include the articles in this thesis. Finally, the journals have given the authorization to publish the thesis after passing the embargo period.

Primarily, Chapter 1 of this document was writing using the paper entitled "*Industry 4.0: Survey from an Integration Perspective" in* the International Journal of Computer Integrated Manufacturing. The other papers are structured in this thesis project as follows:

1.7.1 Chapter 2. Background

This chapter was written from the paper titled "*Industry 4.0: Survey from an Integration Perspective". Consequently, t*his chapter presents the basics to understand the notion of Industry 4.0 and relatives concepts. Firstly, it presents the evolution of the industry towards Industry 4.0, as well as the concepts and technologies related to it. Likewise, an evolution of the IoE concept initially proposed by Cisco is presented in order to allow expanding and generalizing this paradigm. Moreover, it introduces a definition of these 5C levels from our point of view. Furthermore, it presents a discussion of the most commonly used technologies around Industry 4.0 versus the 5C.

1.7.2 Chapter 3. Integrating Multi-Agent Systems and cloud services in intelligent environments

The paper titled "*Fog computing for the integration of agents and web services in an autonomic reflexive middleware"* in the International Journal of Service-Oriented Computing and Applications and the paper titled "*Integración SOA-MAS en Ambientes Inteligentes*" in DYNA are the sources used to write this chapter, presenting the first step regarding the integration problem in Industry 4.0. Notably, this chapter shows how to deal with autonomous communication of MAS and the Cloud Computing paradigm in order to allow intelligent agents to extend their capabilities with cloud components. The solution developed in this chapter is crucial in Industry 4.0, due that MAS and Cloud Computing are two essential components to deploy solutions for Industry 4.0 (see Chapter 2 Section 2.5). In the first place, MAS is used to bring autonomy to the system based on intelligent decisions, while the IoS empowers the system with a vast network of elements deployed in the cloud. Mainly, Chapter 3 presents a State of the Art in MAS cloud-computing integration as the first topic. Next, it presents the MAS-Cloud Integration architecture. Later, it shows how to incorporate

the Fog Computing paradigm to that architecture to avoid issues related to cloud-based systems like real-time and low latency, among other issues. Subsequently, it discusses how this architecture can be integrated within the Industry 4.0 context. Finally, it presents a case study and results that show that this integration architecture for MAS-Cloud computing brings huge benefits to any intelligent environment.

1.7.3 Chapter 4. Autonomic Cycles of Everything Mining for Coordination in the Context of the Industry 4.0

At this point, we decided to move forward in the 5C stack levels and design a framework in order to deal with the integration and interoperability issues regarding the levels of coordination, cooperation, and collaboration. In that sense, the paper entitled "*Autonomic Cycles of Everything Mining for Coordination in the Context of Industry 4.0*", in the International Journal of Industrial Information Integration, constitutes this chapter. Firstly, it presents a start of the art regarding the coordination problem in Industry 4.0. Secondly, it reveals the proposed framework for autonomous integration and interoperability of actors in the Industry 4.0 context. Thirdly, it describes the specification of three autonomic cycles for autonomous coordination in Industry 4.0. Fourthly, it presents a case study that shows how the autonomic cycles are instantiated in a factory to enable the autonomic coordinator. Finally, this chapter shows a comparison of our framework with other related researches.

1.7.4 Chapter 5. Implementing self-* autonomic properties in self-coordinated manufacturing processes for the Industry 4.0 context

This chapter comprises the paper titled "*Implementing self-* autonomic properties in self-coordinated manufacturing processes for the Industry 4.0 context*" in the International Journal of Computers in Industry. This chapter implements one of the three cycles for autonomous coordination designed in Chapter 4. Firstly, the autonomic cycle for self-supervising is detailed methodologically. Secondly, it shows a case study and the instantiation of the autonomic cycle for self-supervising in it. Thirdly, it exposes the results and present some comparison with similar research works.

1.7.5 Chapter 6. Conclusions and Perspectives

This chapter concludes the thesis and outlines the perspectives of this work.

Chapter 2

Background

Contents

2.1 Introduction

In recent years, it has arisen a revolution named Industry 4.0, presented as the integration of new advances in areas such as Cyber-Physical Systems, the Internet of Things and Everything (IoE), Cloud computing, the Internet of Services, Big Data Analysis, Smart Factories, Augmented Reality, among others. Industry 4.0 is not only a new industrial revolution, but also a crucial integration challenge that involves several actors from the IoE, which are people, data, services, and things. This chapter gives a brief review of the evolution of the industry towards the fourth industrial revolution and introduces several fundamental concepts in this context.

2.2 Towards the Industry 4.0

In ancient times of history, all aspects related to production, such as agriculture, transport, or the textile area, among others, were manually carried out, and the development of a product took a considerable amount of time. An essential fact in the

history that came to improve this situation considerably was the emergence of the first industrial revolution.

The first industrial revolution (Industry 1.0) started from the second half of the century XVIII and took until the mid of century XIX. It was driven by the creation of the water pump and the steam engine, which allowed to mechanize the production and to transform the manual production processes into manufacturing processes (Huber & Weiss, 2017; X. Li et al., 2017a; Molano et al., 2017; Santos et al., 2017). In such sense, the manufacturing processes could transform raw materials into products with the help of steam machines, reducing production times considerably.

Subsequently, a second remarkable transformation of the industry, known as the second industrial revolution (Industry 2.0), started. This transformation of the industry began at the end of the century XIX until the beginning of the century XX (X. Li et al., 2017a; Molano et al., 2017; Santos et al., 2017). According to (Huber & Weiss, 2017; X. Li et al., 2017a; Molano et al., 2017; Santos et al., 2017), the elements that drove the second industrial revolution were the electricity and the division of labors. Those elements allowed manufacturing products through assembly lines. Moreover, this revolution drove many changes, like the invention of the internal combustion engine, the discovery of new sources of energy (electricity, oil, gas, among others), and the airplane, among others.

Industry 3.0 started in the 1960s and extended until the beginning of the century XXI (Molano et al., 2017; Santos et al., 2017). This revolution is also known as the Digital Revolution (Santos et al., 2017) because it was centered on the use of the information technology (IT), electronic circuits and the Internet, in order to improve the production processes (Huber & Weiss, 2017; X. Li et al., 2017a; Santos et al., 2017). In this revolution, the Programmable Logic Circuits (PLC) made possible, together with the Industrial Automation (basically, directed by the AI), the integration of automatic machines into the production lines, allowing to reduce human errors and to increase dramatically the development of products and services. Likewise, Industry 3.0 allowed the creation of more efficient, safer, and less polluting means of transport, besides, the use of renewable energy was expanded, and the use of intelligent objects begun.

2.3 Industry 4.0

In recent years, the Industry 4.0, or Fourth Industrial revolution term, was announced at the Hanover Trade Fair, in 2011 (Romero et al., 2017; Santos et al., 2017). However, some authors affirm that the Fourth Industrial Revolution started at the beginning of the century XXI (Molano et al., 2017; Santos et al., 2017). Santos et al. (2017) affirms that this revolution is characterized by the integration of AI solutions within the production machines. Moreover, Li et al. (2017a) affirm that the primary objective of Industry 4.0 is achieving high levels of operational efficiency and productivity. In the Industry 4.0 context, new technologies, such as Cyber-Physical Systems, Internet of Everything, Cloud Computing, Augmented Reality, Big Data Analysis, among others,

are expected to play a crucial role, in order to enable factories to self-organize and self-control, in a distributed and in a real-time way. Notably, that conjunction of technologies has allowed the creation of Smart Factories (Molano et al., 2017; Riel & Flatscher, 2017; Romero et al., 2017), which can autonomously create smart products through smart processes and procedures (Molano et al., 2017).

2.3.1 Smart Factory

A Smart Factory defines innovative production mechanisms, which are suitable for a diversity of applications in different industrial branches, involving modern production technologies. (Liu et al., 2017). Strozzi et al. (2017) affirm that factories become smarter, more efficient, safer, and more environmentally sustainable, thanks to the intelligent combination and integration of production technologies and devices, information and communication systems, data and services, and network infrastructures. Also, Syberfeldt, Danielsson, and Gustavsson (2017) says that the smart factory concept is intended to enable extremely flexible, and self-adaptable production processes, with machines and products that act both, intelligently and autonomously, by implementing concepts such as IoT and CPS.

Moreover, Deloitte Consulting (2017) defines the smart factory as one of the main features of Industry 4.0, which focuses on integrating various industrial devices to establish a networked manufacturing system. Those industrial devices interoperate autonomously in order to achieve the manufacturing goals. Mainly, a Smart Factory is a fully integrated industry (Romero et al., 2017) that combines many technologies, such as 3D Printing, AR, RFID, ERP, IoT, Smart predictive decision support tools, etc. (X. Li et al., 2017a; Strozzi et al., 2017), which use the AI to increase the productivity and efficiency of factories (Molano et al., 2017). Smart factories combine several smart devices that autonomously coordinate, collaborate, and cooperate in order to achieve the production objectives. The main features of a smart factory are (Deloitte Consulting, 2017):

- Connected: The actors involved in the production process can connect and communicate.
- Optimized: the production time is reduced, and the use of human and physical resources is optimized.
- Transparent: new tools are used to support quick, transparent, and consistent decision-making processes, as well as to track the orders appropriately.
- Proactive: early identification of quality issues and anomalies allows self-planning and rescheduling in real-time.
- Agile: adaptable layouts and equipment.

2.3.2 Cyber-Physical Systems (CPS)

CPS was defined by Zanero (2017) as a set of computational and physical interconnected resources, which uses a smart control loop, in order to adapt and improve the autonomy and efficiency of the whole system. In the same sense, Qiu et al. (2017) affirms that CPSs are characterized by strong interactions among cyber components (software) and physical components (devices). Both elements combine AI, automation, and communication technologies tightly, in order to achieve high levels of performance, reliability, efficiency, and robustness in a physical environment (Goossens & Richard, 2017). Likewise, Elattar et al. (2017) define CPS using a service-oriented vision: "a CPS consists of one or more interconnected components or units, where services of each unit are visible to the other units of the system and allow them to cooperate". Jazdi (2014) affirms that a CPS connected to the cloud is often denoted as the "Internet of Things". On the other hand, a CPS is essential in the context of Industry 4.0, because it helps to achieve production goals of the manufacturing processes while improving the effectiveness and efficiency of the entire industry. According to Napoleone et al. (2020), the CPS's characteristics are:

- Integration. It is oriented to join computational and physical processes (Penas et al., 2017).

- Smartness. It is an essential characteristic of CPSs. It allows sensing events, interacting, analyzing the data, making decisions, in order to adapt and change the environment's conditions (Tu et al., 2018).

- Virtualization. Enables the replication of the entire value chain as "digital twins". Virtualization permits analyzing and tracking physical processes, getting feedbacks, and granting the digital continuity of the data about the systems (Sanderson et al., 2018).

- Cooperation and Collaboration. The combination of cooperation, collaboration, and autonomy endows autonomic characteristics as self-organization and self-maintenance of the system (Panetto et al., 2019; Tran et al., 2019).

- Predictability. CPSs must be able to predict the system's behavior, oriented to anticipate unexpected events and failures, to guarantee a continuous execution of the system's processes (Heiss et al., 2015).

- Reconfigurability and adaptability. These characteristics enable the system to dynamically respond to changes and disturbances occurring in the system (Morgan & O'Donnell, 2017).

2.3.3 Cloud Computing

Cloud computing has emerged as a new computing paradigm that offers excellent potential to share networked computing resources. Shila et al. (2017) define Cloud computing as a revolutionary paradigm to deliver computing resources, ranging from

data storage/processing to software, as a service over the network, typically using Internet technologies. Additionally, (Shila et al., 2017; Vaquero et al., 2008) present the US National Institute of Standards and Technology (NIST) referential architecture that has categorized cloud computing into three service models: Software as a Service (SaaS), Platform as a service (PaaS), and Infrastructure as a service (IaaS), which allow users to access software, operating systems, tools, and hardware, as a service over the Internet.

Shila et al. (2017) affirms that the IoS facilitates the organization of various applications into interoperable services, as well as the use of semantics for the understanding, combination, and processing of data and information from different service providers, sources, and formats. On the other hand, Vaquero et al. (2008) affirm that the main target of IoS is to present everything as a service on the Internet, including software applications (SaaS), the platform to develop and deliver these applications (PaaS (Exposito & Diop, 2014)), and the underlying infrastructure (IaaS). In that sense, everything as a service (XaaS (Perera et al., 2014)) refers to delivering anything as a service and includes the vast number of products, tools, and technologies that vendors can deliver to users as a cloud service.

2.3.4 Internet of Things (IoT) and Internet of Everything(IoE)

IoT is related to a network of physical devices and other items, with installed capabilities to allow the connectivity (K. Chen et al., 2017; Mezghani et al., 2017a, 2017b). The authors of (Riggins & Keskin, 2017; Sengupta et al., 2017) affirm that the purpose of IoT is to provide wireless communication to various physical objects that we use daily. On the other hand, Gupta et al. (2016) expand this definition, saying that some of those mobile devices are semi-autonomous or autonomous (smart), and can actuate in their surroundings to provide services to users, who may or may not be in the physical proximity of devices.

Similarly, Cisco defines the IoE as an era in which unprecedented value is created by real-time interaction between actors, including not only things, but also people, processes, and data, all connected to the Internet (D. Lee et al., 2017). (X. Chen et al., 2017; Martino et al., 2018; Shaikh et al., 2017; Yang et al., 2017) introduce IoE as the Internet that connects people, data, processes, and objects, giving connectivity of anything-anytime-anyplace for more intelligent health, smarter energy-efficient cities, smart transportation, among others.

In this thesis project, the definition of Cisco for IoE is extended to include the services dimension, see Figure 2.1. For instance, in IoE, cloud services like storage services, clustering services, among others, are commonly used in order to outspread the limited capabilities of physical devices. Thus, by using Cisco's IoE definition, there is no place for services, because a service is not always a process neither data nor people or things.

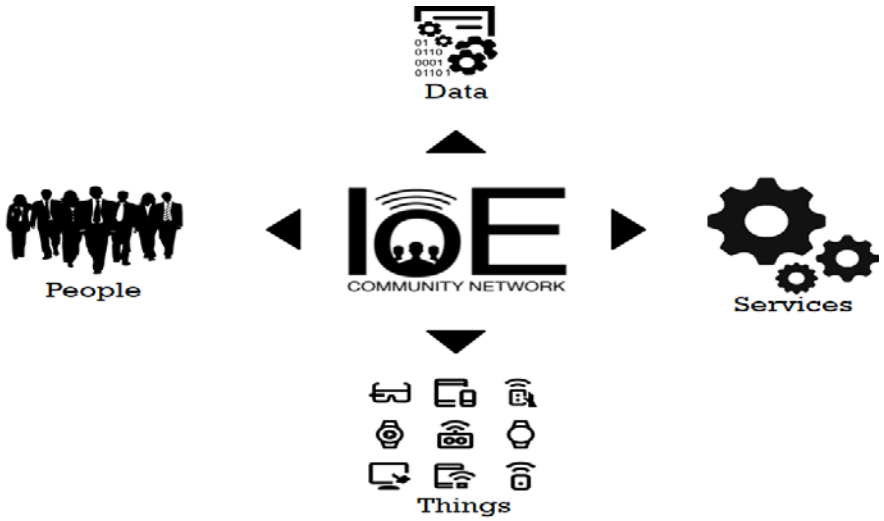

Figure 2.1. Actors of IoE

On the other hand, a process (industrial or not) can be offered as a service using the XaaS concept of the cloud-computing paradigm. In this sense, this thesis project proposes to enhance Cisco's IoE framework by including the service dimension. In other words, IoE is redefined as the interconnection of (see Figure 2.1):

- People: Humans behind a device using a human-machine interface, a wearable device, or social networks.
- 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, among others.
- Services: This is related to the XaaS model of cloud computing, which means anything that can be accessed using a web service interface, like a Database (DBaaS), Knowledge (KaaS), Software (SaaS), Business Processes (BPaaS), among others.

2.3.5 System Integration

Systems Integration is related to link together system components (Auger et al., 2017) like software, hardware, or other systems and sub-systems. Those components interoperate and provide solutions according to their goals (collectives or individuals) (Drăgan et al., 2017). In the context of Industry 4.0, systems are usually integrated using technologies like IoT (Khan et al., 2017; Mezghani et al., 2017a, 2017b), enabling the interoperability of devices and allowing them to connect, communicate, coordinate, collaborate and cooperate. According to Suri et al. (2017), integration is given in three ways:

- Horizontal Integration (inter-company integration): This is based on the collaboration or cooperation between two or more companies to achieve

individuals or common goals (Khan et al., 2017; Suri et al., 2017).

- Vertical Integration (intra-company integration): The vertical integration brings together the system components within an enterprise, like production business process (BPaaS), applications (SaaS) devices, people, data, in order to allow them to coordinate, cooperate or collaborate (Khan et al., 2017; Suri et al., 2017).

- End-to-End Integration: The end-to-end integration mix the digital and real worlds in such a way that real entities can interact with the cyber components of the system. For instance, devices that connect to the network and can send information to the cloud, or people that communicate with the system using an HMI.

Consequently, regarding the system integration concept, it is appropriate to say that this research work proposes an incremental approach that deals with the integration challenges in Industry 4.0. This approach is called the 5C integration levels and consists of solving the issues on the specific layer of the 5C to which they correspond. In that sense, connection issues must be solved in the first level. Next, the communication issues must be dealt in the second level. Once those actors can connect and communicate, the coordination, cooperation, and collaboration issues can be treated according to the integration needs of the manufacturing process. Coordination processes will allow a vertical integration between actors; however, cooperation and collaboration will allow integration in both sides (horizontal and vertical).

2.4 Autonomic computing

IBM presented in 2001 the Autonomic Computing paradigm (IBM, 2004; Lalanda et al., 2013; Parashar & Hariri, 2005; Sterritt & Hinchey, 2005; Vizcarrondo et al., 2012). The purpose of autonomic computing is to provide autonomic properties to systems based on an intelligent control loop known as MAPE (Lalanda et al., 2013). The different levels that composed the autonomic computing architecture are described below (see Figure 2.2)

1. **Resources Management:** Any type of resources (hardware or software) that can be controlled, and may have embedded self-managing attributes.

2. **TouchPoint:** It is the link to sensors and actuators. It is essential to manage the resources.

3. **Autonomic Manager:** It implements the intelligent control loop that automates the tasks of self-regulation/self-management.

4. **Orchestrating Autonomic Managers:** Because an autonomic system can have several autonomic managers that need to work together to ensure the proper functioning of resources, this level provides the communication channel for the coordination between them.

5. **Manual Handler:** It allows humans to configure the autonomic managers oriented to perform its task of self-management, providing for this a man-machine interface.

6. **Knowledge Resource:** It provides access to the knowledge required to perform autonomic management of the system.

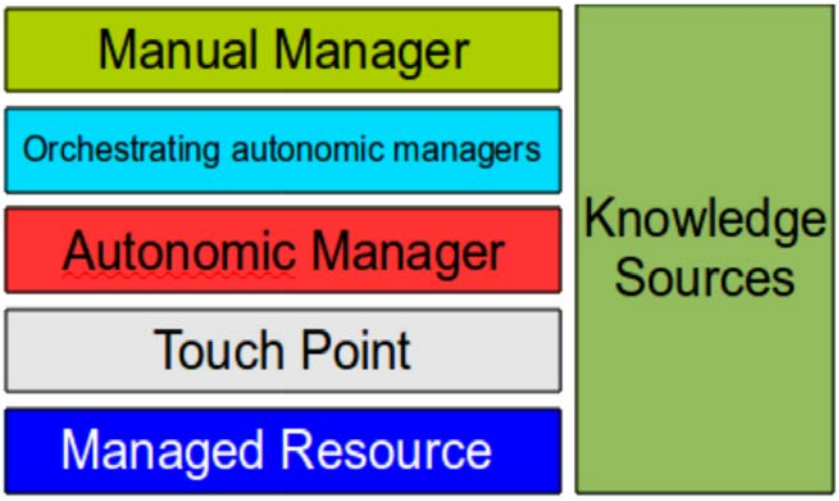

Figure 2.2. Autonomic computing architecture

The main element of the Autonomic computing architecture is the Autonomic Manager because it performs the intelligent control loop (MAPE). The MAPE loop collects, aggregates, and filters data of the managed resource (Monitor phase), provides mechanisms to study complex situations and analyze future situations (Analysis phase), defines the set of operations that must be executed to achieve the system's goals (Planning phase), and provides mechanisms to carry out the plan (Execution phase), see Figure 2.3.

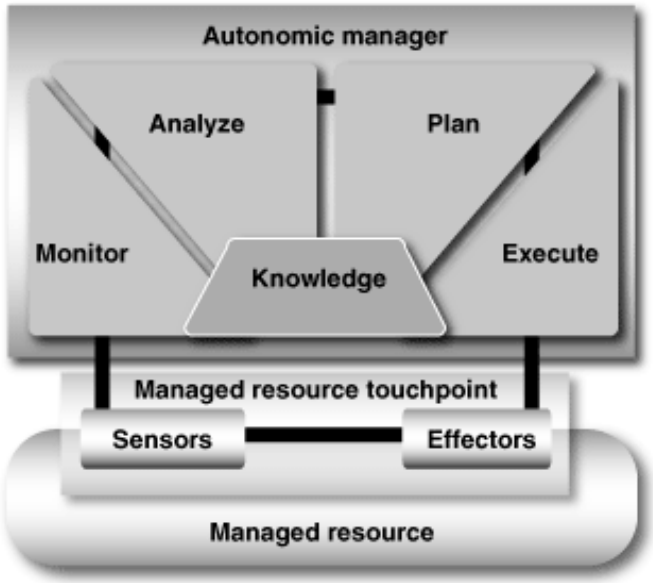

Figure 2.3. Intelligent Control Loop (MAPE)

2.4.2 Autonomic Cycles of Data Analysis Tasks

The main objective of the Data Analysis cycle is to extract useful knowledge from data to allow decision making based on it (Aguilar et al., 2017b, 2017d). In general, an autonomic cycle generates descriptive, identification, and predictive models, among others, based on data, in order to guide the decision-making processes in the system.

On the other hand, an "Autonomic Cycle of Data Analysis Tasks" is a concept defined in (Aguilar et al., 2017b, 2017d), which consists of a set of Data Analysis tasks that acts together, in order to achieve an objective in the process that they supervise. The tasks interact with each other and have different roles in the cycle. The roles are: Observing the process, analyzing and interpreting what happens in it, and making decisions, which allow reaching the objective for which the autonomic cycle was designed.

The integration of Data Analysis tasks in a closed-loop allows solving complex problems. In this sense, it is essential to integrate the Data Analysis tasks in a coherent way to generate useful and strategic knowledge for the achievement of the objectives. The detailed description of the roles of each task is (Aguilar et al., 2017b, 2017d):

- Monitoring: These tasks are in charge of observing the supervised system. They must capture data and information about the behavior of the system. Besides, they are responsible for the preparation of the data for the next steps: preprocessing, selection of the relevant features, etc. See (Aguilar et al., 2017d) for more details.

- Analysis: Corresponds to a set of tasks to interpret, understand, and diagnose what is happening in the monitored system. These tasks allow building models of knowledge about the dynamics observed by the cycle in order to understand the system needs and to make future decisions.

- Decision making: These tasks define and implement the necessary actions based on previous analysis, oriented to improve, detect failures, among others, in the supervised system. These tasks impact the dynamics of the system in an intent to improve it. The effects of these tasks are evaluated in the monitoring and analysis steps, restarting a new iteration of the cycle.

The concept of “Autonomic Cycles of Data Analysis Tasks” has been used in different domains. For instance, it has been used in the context of Smart Classroom (Aguilar et al., 2017d; Sanchez et al., 2019), Smart cities (Aguilar et al., 2016, 2017b, 2019), among others.

2.4.3 Methodology for the development of Data Mining applications (MIDANO)

MIDANO is a methodology that guides the development of the autonomic cycle of data analysis tasks and consists of three phases (Aguilar et al., 2017a; Pacheco et al., 2014; Rangel et al., 2013). Figure 2.4 details each phase of this methodology.

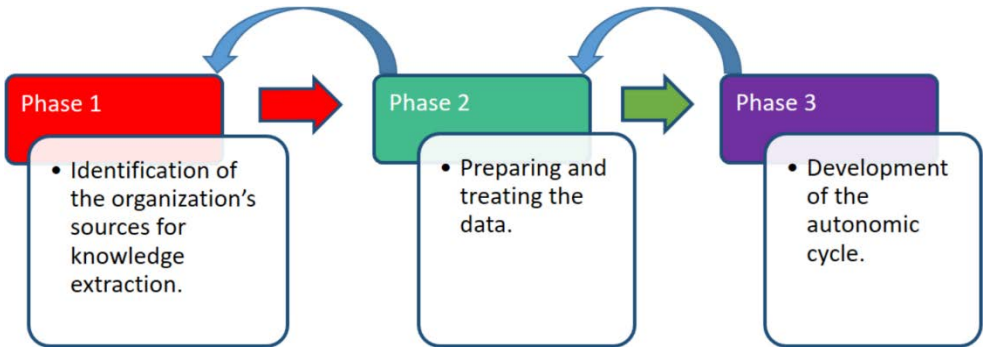

Figure 2.4. MIDANO's methodology (Rangel et al., 2013; Pacheco et al., 2014; Aguilar, Aguilar, et al., 2017; Lopez et al. 2019)

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

Phase 2: This phase is based on an ETL process, 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. This task allows creating the required knowledge models, such as predictive models, descriptive models, etc. Also, this phase ends with the implementation of a prototype of the autonomic cycle.

2.4.4 Everything mining

Everything Mining is the application of any mining technique to any actor, such as the process mining, big data mining, service mining, things mining, and people mining (sentiment analysis, opinion mining, etc.), with the primary goal of getting a better understanding of the system and learn from it. Moreover, Everything mining allows the system to get useful information from all actors involved in the system, such as data, things, people, and services. Additionally, that information is used to create the knowledge bases that are used by the autonomic cycles to make decisions. It means that the Everything mining technique is useful for the aggregation of new self-* properties to the system.

Consequently, everything mining is related to the concept of mining of the heterogeneousness or multi-modal Big data sources. Thus, everything-mining includes not only data mining technology, but also process and service mining (mining from events rather than only data), or whatever another kind of mining technology.

2.4.5 Big Data Analytics

In Industry 4.0, the integration of multiple manufacturing processes has generated an overflow of data from different sources. This data requires new approaches for its management (Khan et al., 2017). In this sense, Big Data deals with this problem in

production processes, by pre-processing the data generated mainly by sensors, looking for insights and knowledge that allow the humans involved in the production process to make better decisions. In general, Big Data can is defined using the five "V" as follows (Chang & Wills, 2016; Jirkovský et al., 2017; Obitko & Jirkovský, 2015):

- Volume refers to the large volumes of information that are daily generated.
- Velocity refers to the speed of how the data are produced and must be processed to meet the demands.
- Variety refers to the different types of information formats, whether structured and unstructured.
- Veracity refers to the reliability of the data.
- Value refers to the meaning of the data in the operational context, that is, what is the real benefit that can be derived from them.

Consequently, Big data analytics allows the collection and analysis of a large number of data from different sources in order to support decision-making (Pierdicca et al., 2017). It is fundamental in the context of Industry 4.0 to identify useful patterns, production models, (Jirkovský et al., 2017; Santos et al., 2017), as well as to ease the cleaning, formatting, transforming, and processing of the data (Khan et al., 2017). Mainly, techniques like machine learning, text mining, data mining, process mining, service mining, semantic mining, and massively parallel data processing like map-reduce and Hadoop, cloud computing, databases oriented to graphs and events, databases without schema, etc., are necessaries for the analysis of the vast amount of data generate by industries.

2.5 Integration Technologies used in Industry 4.0

A summary of the commonly used technologies that allow the integration and interoperability of actors in Industry 4.0 is shown in Figure 2.5.

At the connection level, it can be seen that the technologies used to allow actors to get in touch between them are: Augmented Reality, Industrial Internet of Thing, Semantic Web, and TCP/IP. It means that connection not necessarily indicates a network connection; it implies that actors can contact each other. E.g., Augmented Reality makes it possible to connect people with other actors of a production process; however, it can also allow communication or coordination.

Besides, as it was discussed in section 1.5, communication was the level most studied by researchers in past years. It explains the variety of technologies used at that level, as shown in Figure 2.5. Starting from low-level protocols like TCP/IP, medium level protocols like FIPA, until more high-level protocols, such as the service-oriented protocols. Furthermore, technologies as Big Data Warehouse allows communication

indirectly by letting actors getting and sending information through data.

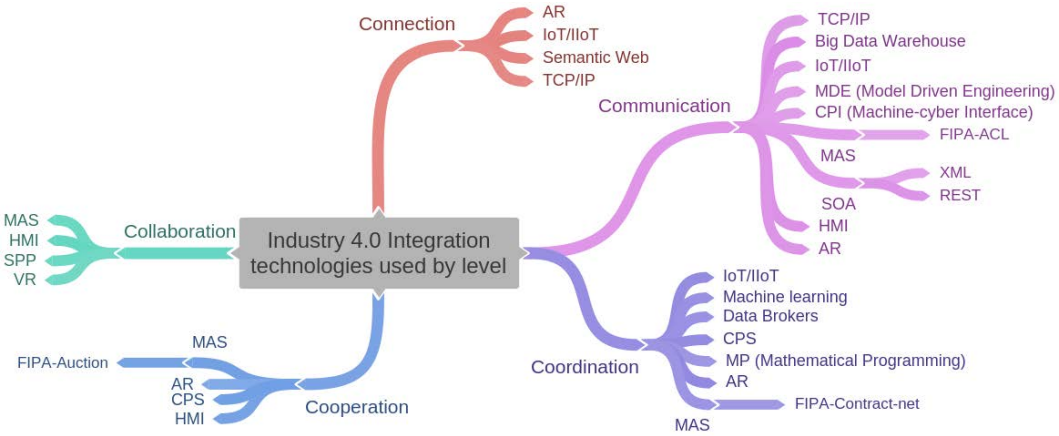

Figure 2.5. Commonly used integration technologies by level.

Accordingly, at the Coordination, Cooperation, and Collaboration levels, it can be noticed that the techniques are more related to AI. Maybe it is due that on those levels, actors need to incorporate negotiation and convergence protocol, in order to resolve conflicts in the execution of their task. Moreover, actors must deal with the complexity of planning and decision-making, and AI helps to solve those challenges.

2.6 Summary.

Industry 4.0 is a concept still in development that arises from the integration of technologies, such as AI, Smart factories, CPS, Cloud Computing and IoS, IoT and IoE, Systems Integration, HMI, and Big Data Analysis, among others. In that sense, Smart Factory is an essential feature because it endows autonomy to the production process, allowing it to self-configuring, self-supervising, self-healing, among others. In a Smart Factory, devices interoperate autonomously intending to achieve manufacturing goals, taking care of efficiency, and resource usage. Besides, CPS is another crucial technology used in Industry 4.0 to bring autonomy to production processes, due that CPS uses a smart control loop that allows adapting and improving the efficiency of the whole system. In this case, a CPS lets services and devices interoperating, in order to achieve production goals, as well as improving the effectiveness and efficiency of the entire industry.

IoT/IoE is used as an integration layer, which not only allows the actors of the production process to communicate and interoperate but also to extend their capabilities by using services deployed through the cloud computing paradigm. In this sense, devices, people, data, and services can connect and communicate using the standards provided by IoS and cloud computing. This characteristic is essential to promote autonomous processes for coordination, cooperation, and collaboration, in order to drive the interoperability of actors and allow them to achieve, intelligently and efficiently, collective and individual goals.

As can be seen, Industry 4.0 is fully integrated, so system integration is a crucial

aspect in this context because it allows integrating recent technologies and putting them to work together to increase the autonomy of the production process. In the same sense, Modern Human-Computer Interactions is very useful in Industry 4.0, because it allows people to be integrated into the manufacturing process transparently. Technologies like wearables, AR, and others, allow people to interoperate with other actors in a more intuitive way, increasing cooperation, collaboration and coordination of those actors to improve, in this way, the production processes, reduce waiting times, increase security and save costs, which are valuable characteristics in the Industry 4.0. Finally, Big Data Analysis is essential in Industry 4.0, because it allows dealing with the heterogeneity of data and actors by pre-processing large volume of data in looking for knowledge, identifying useful patterns, production models, among others.

In summary, a traditional industry cannot be turned into the Industry 4.0 concept only by integrating all the technologies described above. Industry 4.0 needs to endow autonomy in the production process. That means, not only autonomous interoperability of actors, autonomous decision-making, autonomous negotiation, etc., but also, self-configuring, self-healing, and self-supervising, among other self-* properties that bring autonomy to production processes. In that sense, the autonomous coordination, cooperation, and collaboration processes will be useful to let actors organizing, conjointly act and efficiently drive the production processes.

Chapter 3

Integrating MAS and cloud services in intelligent environments

Contents

3.1 Introduction

Service-Oriented Architecture has emerged as a dominant architecture for interoperability between applications by using a weak-coupled model based on the flexibility provided by Web Services, which has led to a wide range of applications, known as cloud computing. On the other hand, Multi-Agent System is widely used in the industry, because it provides appropriate solutions to complex problems proactively and intelligently. Specifically, Intelligent Environments (Smart City, Smart Classroom, CPSs, Smart Factories, and Industry 4.0, among others) obtain significant benefits by using both architectures, because MAS endows intelligence to the environment. At the same time, SOA enables users to interact with cloud services, which improve the capabilities of devices with services deployed in the cloud.

Additionally, the fog computing paradigm extends the cloud computing paradigm to be closer to the things that produce and act on the intelligent environment, allowing them to deal with issues like mobility, real-time, low latency, geo-localization, among

other aspects. In this sense, in this chapter, we present a middleware, which not only is capable of allowing MAS and SOA to communicate in a bidirectional and transparent way but also, it uses the fog computing paradigm autonomously, according to the context and the system performance. Furthermore, we introduce a scenario oriented to show the applicability of our proposal in the Industry 4.0 concept. In consequence, the solution presented in this chapter intents to solve integration challenges at the level of communication of the 5C stack levels.

3.2 State of the Art in MAS-Cloud computing integration

This section shows a brief description of the works that propose the integration of MAS, fog-computing, and cloud computing. The first subsection describes those works that address the issue of integration or combination of MAS with the fog computing paradigm, while that the second subsection presents those works that combine MAS with SOA platforms or with the cloud-computing paradigm.

3.2.1 Integration MAS and Fog Computing.

The Cloud of Things (CoT) platform was presented by Amadeo et al. (2017), intending to deal with some challenges in the smart home domain. Also, Amadeo et al. (2017) present two revolutionary concepts: the Information-Centric Networking (ICN) and the Fog Computing. The proposed ICN-iSapiens is a three-layered architecture that uses an intermediate (Fog) layer, consisting of smart home servers (HSs). This layer is introduced between the physical world and the cloud, to support real-time services and hide the heterogeneity of the Internet of Things (IoT) devices (Riggins & Keskin, 2017; Sengupta et al., 2017). The ICN Physical Layer includes all the edge devices (EDs), which deploy sensing and automation tasks in the smart home. EDs are usually single devices, like temperature, motion sensors, or light actuators. The Intermediate Fog layer includes the HSs, which implement the application logic to monitor and control the house according to:

(i) The user preferences

(ii) The inputs from stakeholders (e.g., service providers and regulation entities)

(iii) The dynamic context-related factors (e.g., the energy market price).

The HSs interact with the EDs through ICN, and with the remote cloud, through standard Internet protocols. Besides, the Named Data Networking (NDN) interacts with cloud and fog computing to the benefit of high-quality, flexible services, and global reachability. Each ICN-ED is represented in the Fog Layer as a virtual object (VO), which is a high-level standardized description of the device's functionalities. VOs expose EDs by hiding their heterogeneity in terms of technological and networking details and make their resources easily accessible to Agents, which, in turn, perform the application logic. Agents use VOs methods to pull monitoring and action tasks, and to be asynchronously notified about some events, i.e., when a resource value changes. Besides, they may subscribe to complex events over groups of functionalities.

Therefore, instead of transferring data to a central processing unit, ICN-iSapiens transfers processes (Agents) towards the EDs. The Remote Layer includes a remote Cloud platform, which addresses all those activities that cannot be executed by the HSs, e.g., tasks requiring high computational resources or long-term historical data. The data analysis executed in the Cloud-computing can be used for different purposes, including (i) optimizing the Agents' operations and their behavior, or (ii) to support the demands of external consumer applications (e.g., collected data can be used by energy companies for reliable forecasting).

Peng, Xu, Gates, Cui, & Lin (2017) present a study of the collaborative, self-adaptive configuration management of virtual machine resources in a multi-agent organization structure, with virtual machines as the unit of granularity. The multi-agent organization structured with virtual machine clusters consists of four types of agents. The vms-agents and apps-agents are collaborative agents. One vms-agent and one apps-agent are present in each virtual machine cluster. The vms-agent in the virtual machine cluster is responsible for implementing the virtual machine resource configuration requests issued by the vm-agent. Also, the vms-agents propose dynamic parameter configurations for the relevant application systems/components that are operating on the virtual machines. Besides, vms-agents sends real-time utility information on various types of resources to the apps-agent. Then, the apps-agent is responsible for implementing the application system/component parameter configuration requests issued by the com-agent. The apps-agent proposes dynamic resource configurations for the relevant virtual machines on which the application systems/components are operating. Also, it sends various types of real-time information regarding the performance of the application systems/components (e.g. response time and throughput) to the corresponding vms-agent. The vm-agents and com-agents are intelligent agents that can adapt themselves to the cloud computing environment. One vm-agent and one com-agent are deployed in each virtual machine and each application system/component. The vm-agent learns the optimal decision regarding the virtual machine resource configuration, and monitors and provides feedback on various types of virtual machine resources in real-time. Then, the com-agent learns the optimal decisions regarding the application system/component parameter configuration, and monitors and provides various types of feedback on application system/component performance in real-time. Experimental results show that their method improves the efficiency in resource utilization, and accounts for the interests of both cloud users and providers while ensuring the maintenance of the application service level agreement.

Similarly, Giordano, Spezzano, & Vinci (2016) combine agent technology with the concept of Fog computing to design control systems based on the decentralization of control functions over distributed autonomous and cooperative entities that are running at the edge of the network. Also, they present Rainbow, an architecture that allows the development of smart city applications. Rainbow is a three-layer architecture designed to bring the computation as close as possible to the physical part. The bottom layer is devoted to the physical part. It encloses sensors and actuators,

together with their relative computational capabilities, which are directly immersed in the physical environment. In the Intermediate layer, sensors and actuators of the physical layer are represented as VOs. VOs offer transparent and ubiquitous access to the physical part due to a well-established interface exposed as API. Also, VOs allows agents to connect directly to devices without caring about proprietary drivers or addressing some kind of fine-grained technological issues. Besides, all the devices are appropriately wrapped into VOs that, in turn, are enclosed in distributed gateway containers. The computational nodes that host the gateways represent the middle layer of the Rainbow architecture. The upper layer of this architecture concerns the cloud part. This layer addresses all the activities that cannot be executed appropriately in the middle layer, for instance, algorithms needing knowledge, tasks that require high computational resources, or when the historical data storage is mandatory.

In contrast, all tasks requiring real-time access to the physical part, are executed in the middle layer. Communication between nodes, the physical part, and the nodes in the cloud occurs through messages exchange. Agents located in the cloud nodes act as intermediaries between the Rainbow, MAS, and cloud analytics services. The features and capabilities provided by the Rainbow framework are shown by running intelligent algorithms, in order to realize CPS applications (Jazdi, 2014; Zanero, 2017), owning properties such as additivity, fault tolerance, and self-reconfiguration, among others.

Finally, Mohamed et al. (2017) discuss how the service-oriented middleware (SOM) approach can help to solve some challenges related to developing and operating smart city services, using CoT and Fog Computing. They propose a SOM called SmartCityWare, for the integration and utilization of CoT and Fog Computing. SmartCityWare services and components involved in the smart city applications are accessible through the service-oriented model. The primary purpose of SmartCityWare is to provide a virtual environment to develop and deploy smart city applications. SmartCityWare consists of a set of services and a multi-agent runtime environment. All functions of the SmartCityWare are viewed as a set of services that can be used to build and support the execution of different smart city applications. These services are classified into core services and environmental services. Core services are developed specifically for the core operations of the SmartCityWare, such as the broker, security, service invocation, and location-aware services, to provide overall control for the whole system. Environmental services allow accessing services provided by one or more cloud service providers, services provided by multiple distributed fogs, and services provided by multiple IoT devices, including sensors, WSN (Faheem & Gungor, 2017), actuators, cameras, cars, robots, etc. Fog services can be used to control, processing, storage, communication, streaming, configuration, monitoring, measurement, and management services. SmartCityWare services can be used by smart city applications available on the cloud, fog, or IoT devices, such as a car asking for a specific service from a smart city application existing on the cloud.

Moreover, the SmartCityWare middleware infrastructure utilizes software agents for providing flexible and expandable middleware services, or high-performance distributed service-oriented environments. The main functions of the agents are to

deploy, schedule, and support the execution of the service codes in different fogs. Also, to manage, control, monitor, and schedule the availability of resources on a single fog, or a set of related distributed heterogeneous fogs. For the experiments, they used three computers; one represents the cloud and two represent fogs. Besides, they used WAN emulators among the machines, to introduce the effects of using long distances and the Internet to connect them.

3.2.2 Integration MAS-SOA or MAS-Cloud computing

This subsection presents those studies that combine MAS with the cloud computing paradigm or some type of software-oriented architecture. In this sense, Archimède, Memon, & Ishak (2017) propose Semantic-SCEPSOA, which is an architecture that combines three kinds of technologies. In the first place, SOA facilitates the identification and relationships of actors on the internet, allowing technical and operational interoperability by gathering functionalities of enterprise applications. In the second place, the ontology technology facilitates the understandings of the information exchanged. In third place, a multi-agent model is used to create the project's plan. The semantic interoperability strategy, necessary to ensure a better understanding and interpretation of the exchanged information between heterogeneous systems, is based on ontologies too. Reasoning mechanisms are needed by customers and producers to transform the data stored in the global ontology into data expressed according to the local ontology, and vice-versa. S-SCEPSOA is organized around three kinds of actors (SOAregister, SOAproducer, and SOAconsumer). The SOAregister contains information about the services, the SOAproducer is in charge of offering services, and the SOAcustomer consumes the services by invoking them at the corresponding SOAproducer. The interaction between the different actors is achieved via messages. Also, the management process is achieved by three steps. The first step is the identification of partners, which concerns the service publication and discovery phases in the SOA context. The second step deals with the instantiation of SCEP (supervisor, customer, environment, and producer) components on the SOAcustomer side, and connection to SOAproducers selected in the previous step. In the third step, the SCEP agent gets from the customer database (CDB) the projects' description and creates a shared environment, as well as the customer and ambassador agents. One customer agent is created for each project. Similarly, one ambassador agent is created for each SOAproducer. The third/last step concerns interactions and cooperation between SOAcustomer and its SOAproducers, through the SCEP instantiation, to manage multi-site production projects.

Al-Ayyoub et al. (2015) present a dynamic resource provisioning and monitoring (DRPM) system, which is a MAS, to manage cloud provider's resources while taking into account the customers' quality of service requirements, as determined by the service-level agreement (SLA). The provider's resources include a set of data centers, each with a large number of physical machines. Each physical machine has specific characteristics (in terms of CPU, RAM, storage, and bandwidth) to host multiple virtual machines (VMs). Moreover, it contains sensors to measure the utilization of its resources and the execution of the customers' applications on it. The application layer consists of a set of customers having several jobs that possibly spans multiple VMs.

Each job is associated with specific performance goals specified in the SLA (e.g., time to complete their tasks, number of requests for specific periods). The multi-agent components include a global utility agent and a set of local utility agents. While the global utility agent has the classical role of the "central broker" that allows it to manage all the system resources, the local agents are assigned to each customer to improve the resource utilization without causing SLA violations. Based on a regression analysis of its history, the local agent can estimate the amount of resources that will be used without causing SLA violations. After reformulating the requests, the local agent sends the new requests to the global agent, which is responsible for the actual provisioning of the resources, by communicating with the supervisors of the physical machines in the different data centers under the cloud provider's control.

On the other hand, García et al. (2014) propose a model for automatic construction of business processes, called IPCASCI (Intelligent business Processes Composition based on multi-Agent systems, Semantics and Cloud Integration), in order to facilitate the business process construction on cloud computing environments. It deals with the development of a proposal that simplifies the creation of such processes, in the form of web services, from another semi-automatic functional service. A multi-agent architecture guides the business process construction based on virtual organizations, which can implement the intelligent behavior needed for process management by using ontologies. The components that integrate the architecture are:

1. The cloud system. It is the Platform of cloud computing on which the proposal is supported. The platform provides an environment for running and store the data.

2. Web services. It means web services existing in the architecture that will be used for web service composition processes.

3. UDDI register. It is the Universal Discovery Description and Integration system where the web services are registered. This component allows open access to the web services of the architecture. Also, it allows reusing the web services implemented in the architecture.

4. Functionalities to make discovery and composition of web services. This process is addressed by analyzing the semantic content introduced by the user in order to structure it in computable items.

5. Ontologies. This component models the semantic knowledge that can be included in the web services.

6. BPEL (Business Process Execution Language) (Decker et al., 2007; Fu et al., 2004). This file stores the composition of the web services that meet the requirements indicated by the user to obtain the desired solution.

The discovery and composition of services are carried out when the user introduces the requirements of the web service. Later, an automatic search (made in syntactic and semantic form) for the web services that include the requirements of the module is performed. Next, the Business Process Diagram (BPD) is displayed by the assistant

software to the user. Finally, from the BPD, a BPEL is created in such a way that the services specified by the user are built, and can later be invoked.

Notably, the research works presented above do not directly deal with SOA-MAS integration, but instead, each agent must apply the necessary message transformations to be able to access cloud services. In this sense, Fuksa (2014) developed a library called JADE Gateway Library, which acts as a bridge between a MAS and the services that are deployed in an Enterprise Services Bus (ESB). The library is composed of two fundamental elements: the first one is the Jade-Gateway, which is responsible for managing the messages received from and to the ESB. The second one is the Gateway Agent, whose main objective is to retransmit the messages between the MAS and the ESB. When an agent wants to consume a web service, it must send a message to the GatewayAgent, which in turn relays it to the ESB. On the other hand, when a web service needs to send a message to an agent, it must contact the EntryPoint ESB Service, which passes the message to the Jade-Gateway and then to the GatewayAgent, which will relay it to the requested agent. During the communication process, the Jade-gateway must perform SOAP-ACL and ACL-SOAP transformations, so that each part receives the message in the appropriate language. In the same way, Jade-gateway solves the problem of synchronous communications integration of web services with asynchronous communications of a MAS, blocking the GatewayAgent thread until obtaining the MAS response.

Finally, Pinto (2014) proposes SoMAS (Service-oriented MultiAgent System) as a service-oriented architecture, where both agents and web services use SOAP as a communication language. In the SoMAS architecture, agents publish their services in the UDDI, by sending the Web Service Description Language (WSDL file corresponding to the service offered. When an agent or web service requires the use of a particular service, it must be looked up in the UDDI, and once found, the communication is established. This architecture leaves aside the FIPA standard since the system does not use a Directory Facilitator (DF) neither the Agent Communication Language (ACL). The author indicates that both communication interfaces can be used (SOAP and ACL), but it represents additional work.

Mainly, the works shown in this section combine MAS-Fog or MAS-SOA paradigms. However, they do not allow the combination of the three paradigms transparently, such that agents can communicate with services naturally, and the fog computing paradigm can be enabled conveniently. In this sense, works that deal with the MAS-SOA integration issues directly (Fuksa, 2014; Pinto Pereira, 2014), did not make use of the fog paradigm, so they are not able to solve several issues of cloud computing, such as mobility, real-time, low latency, etc. Besides, these works have weaknesses, due that they are not FIPA-compliant, or not allow bidirectional communication of agents and services. In this chapter, we propose a new architecture to solve those issues. Also, this chapter studies the performance of the incorporation of the fog-computing paradigm intending to allow agents not only to communicate with services naturally but also to use the fog computing paradigm autonomously.

3.3 Proposed MAS-Cloud integration architecture

Figure 3.1 shows the architecture for MAS-SOA integration. This architecture allows any agent-based system to consume cloud services through conventional proxy agents (WSA), and expose the capabilities of the agents as a service, through the use of service-oriented proxy agents (WSOA).

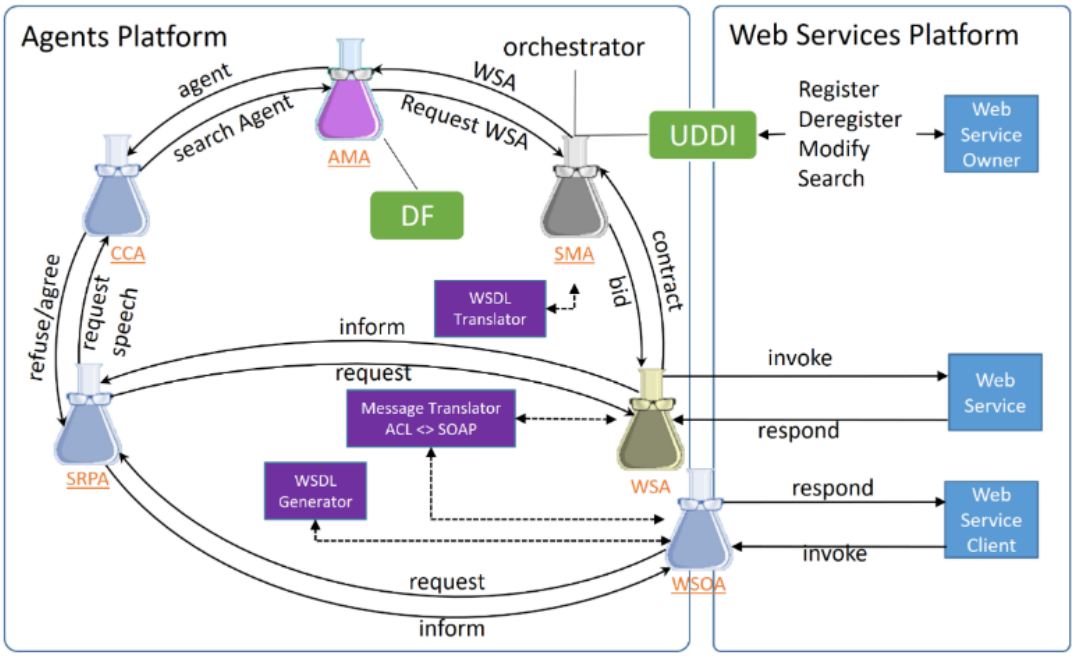

Figure 3.1. Proposed SOA-MAS integration architecture.

The proposed architecture is a mix of the conventional gateway agent approach and the service-oriented agents (SoMAS) approach. Figure 3.1 shows the agents involved in the integration process. The SMA agent is responsible for controlling the UDDI and everything related to the registration/deregistration, modification, and discovery of web services. Each time a web service is registered in the UDDI, a WSA that characterizes that web service is also registered in the SMA, using the "WSDL Translator" module. On the other hand, the WSOA has the role of serving as a façade for agents so that their capabilities can be offered as services in the cloud. Consequently, SRPA represents any agent that might need to invoke a Web Service (in this case, it will be called RA, see Figures 3.2 and 3.3); or exposes its functionalities as web services (in this case, it will be called SA, see Figure 3.4).

Unlike what was proposed by (Greenwood et al., 2005; Greenwood & Calisti, 2004; Shafiq et al., 2006), the WSA agents are not registered in the MAS' DF, because the element in charge of administering the WSA is the SMA agent. For its part, the AMA agent is in charge of controlling the DF, where the agents are registered. When an agent needs to discover another one, AMA searches first in the DF. If it finds the agent, it returns his reference; otherwise, if it does not get the agent in the DF, it sends the requirement to SMA (since it might be requiring a WSA). SMA will search in its directory, and if the agent is found, it returns the WSA that serves as a proxy to consume the required web service. In this way, an agent can discover both other agents

(registered in the DF), and the web services available in the environment (registered in the UDDI). Figure 3.2 shows the steps followed in this process. Orange arrows (messages 1-4) represent the steps needed to discover the required web service. In contrast, green arrows (messages 5-8) show how the requested service reaches the agent that made the request (RA).

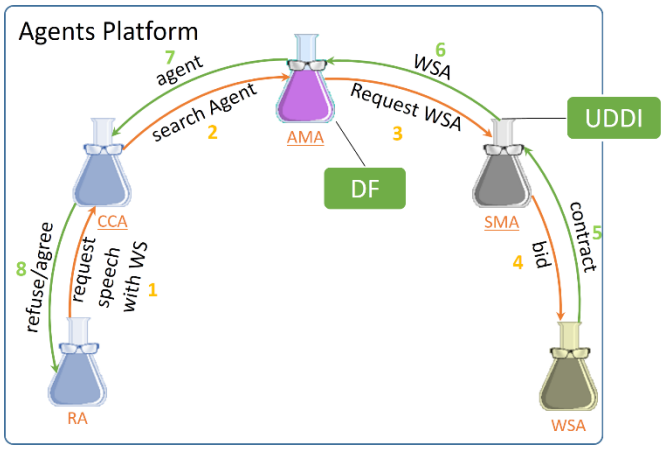

Figure 3.2. Discovering of web services.

Once that RA has the reference to the WSA that characterizes the service needed, it can initiate the communication process to consume this web service. This process is detailed in Figure 3.3. Initially, RA will make the requirement to consume the web service, sending an ACL message to WSA (1), WSA will transform the ACL message to SOAP using the "Message Translator" Module (2), and will relay the message to the web service endpoint (3). If it is asynchronous communication, WSA must block its thread until it receives a response from the web service. As soon as the response is received (4), WSA transforms the message from SOAP to ACL (5), using the "Message Translator" module again, and forwarding the information (6) to the requesting agent.

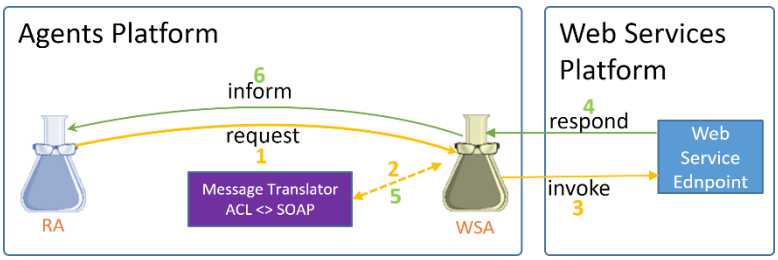

Figure 3.3. Invoking web services.

On the other hand, as stated before, WSOA agents (see Figure 3.1) serve as a façade to expose the capabilities of MAS agents as web services. These agents are automatically created when the agent they serve is registered in AMA (One WSOA is registered in the AMA for each system agent). The WSDL file of each WSOA is also registered in the UDDI, using the "WSDL Generator" module. In this way, the external web service can discover the WSOA (and by consequence, the agent to which it serves) performing a search in the UDDI. The services offered by each WSOA are deployed on a web server, or through a framework such as Apache CXF (2008) or the IoT/IoE platform. Figure 3.4 details both the process of discovery of the WSOA by a web service

(1) and the process followed to invoke the functionalities exposed by the agents of the MAS platform.

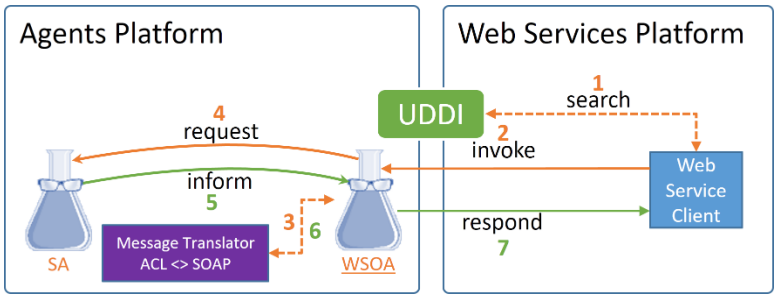

Figure 3.4. Invoking agents' functionalities.

As soon as the "Web Service Client" found the WSOA agent that represents the required agent (SA) in the UDDI, it can invoke the functionalities of that agent as a service. The web service client sends a SOAP message to the WSOA (2). The WSOA takes the requirement and transforms it into an ACL message (3), using the "Message Translator" module. Then WSOA sends the ACL message to the agent SA (4). Due to the asynchronous nature of agents' communications, WSOA must wait until a response from SA is received. Once that it has the information, the ACL message is transformed into a SOAP message (6) through the "Message Translator," and forwarded to the "Web Service Client" (7) that started the request initially.

3.4 MAS-SOA Fog component.

The architecture proposed in Section 3.3 uses the Internet of Services (IoS)-paradigm to allow the interoperability of agents and services. Consequently, agents can invoke services deployed in the cloud in order to extend the capabilities of the physical entities with cloud-computing services. Additionally, web services can invoke agents' functionalities too.

Notably, the agents in the presented architecture are not capable of dealing with some issues related to cloud platforms, smart environments, and IoT, like latency, real-time, location-aware, among others aspects (Bonomi et al., 2012; Mahmud et al., 2018). In this way, we have used the Fog Computing paradigm to deal with these issues, and in this research, we must analyze the benefits of such integration.

Figure 3.5 shows the Fog Component. Each time any agent of the platform needs to invoke a service, it follows, as indicated in Figures 3.2 and 3.3. However, as soon as the WSA is discovered, it contacts the Fog agent (FogA) in order to know where the service will be invoked (fog or cloud). The FogA uses the information collected by the System monitor agent (SMonA) and a Fog Ontology (describe in Aguilar et al. (2017e)), to determine, based in the current context and the system requirements, if it must use the cloud or the fog. This information is returned to the WSA, which will act accordingly.

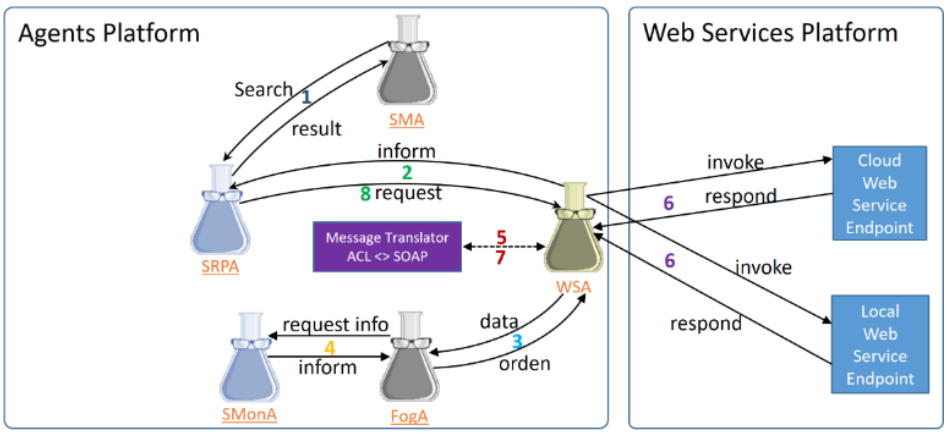

Figure 3.5. Fog interaction between agents.

Furthermore, in the Fog ontology, a weight is assigned to each concept such that the FogA can know what elements have a high priority to be processed locally. Moreover, this information is combined with context information allowing agents to identify what concepts of the fog-ontology will be used. In contrast, the information collected by SMonA about the load level of each agent is precious, in order to determine if the fog elements can process the actual requirement or not. Consequently, all this information helps the FogA to decide what data will be processed locally or in the cloud. The rules that use the FogA to make the decisions are:

- If the SMonA informs that the local services and agents are overloaded, the FogA will require more power of processing in the cloud.

- Otherwise, if the ontology set the concepts with a high weight (according to the context and the domain), and if the SMonA agent informs that the agents or local services are in the capacity to process the data, the FogA decides to process the information locally.

As soon as the FogA decided whether to process the data, the WSA translates the ACL message into a SOAP message and invokes the service using the information returned by the FogA. When the WSA receives the result from the web service, it translates that result in an ACL message and sends the result to the requesting agent. In that sense, the FogA helps to deal with problems of quality of services, high latency, and real-time related to cloud-based systems. In summary, the FogA combines real-time information of the system, with context information defined in the Fog Ontology to make decisions.

3.5 MAS-Cloud Integration in the Industry 4.0 context

In this section, we describe how our approach can be extrapolated to the Industry 4.0 context in order to solve not only the integration of MAS and Cloud-computing but also the issues related to latency, real-time, mobility, among others.

In the Industry 4.0 context, several authors proposed the use of MAS (Huang et al.,

2017; D. Li et al., 2017; Romero et al., 2017; Wang et al., 2016a) to deal with the complexity of planning and negotiation challenges (see section 2.5), as well as the creation of autonomous processes for coordination, cooperation, and collaboration. In this sense, this research can bring considerable benefits to the integration process in a production environment with cloud technologies, which at the same time need to solve latency and real-time issues, among others. To illustrate this idea, we present the next scenario.

Suppose that a company has several devices on an assembly line (see Figure 3.6), where smart devices control the production process. Likewise, some consumers place orders to request customized products. The customer requires to have the product finished at a specific date. It means that the company must accept the elaboration of the product prudently to not incur in production delays. Also, the smart products will coordinate their self-production. In this case, the company requires several mechanisms for integration and interoperability of actors in order to manage the production autonomously. It means that this factory needs the incorporation of 3C processes (coordination, cooperation, and collaboration) (Sanchez et al., 2018a) in the production line. Essentially, this factory has the next requirements.

1. Smart products will coordinate their self-production. To do this, they are needed some appropriate *coordination mechanisms*, directed by the smart product, in such a way that in each phase of the production process, the necessary elements are added to the product according to their specific requirements, which may change from one smart product to another.

2. In the same way, the physical elements (things) of the smart factory must use *cooperative mechanisms* in order to allow them to carry out the production process in an efficient manner. Each actor has particular objectives; for example, the objective of the assembly belt is to transport products from an original place to a specific destination, knowing that there are multiple origins and destinations. The objective of a robotic arm might be to add a layer to the final product, and so on. In this way, cooperation between all the actors will allow each actor to reach their goals to allow the final products to be created properly.

3. The objective of the whole production process is to produce smart products efficiently, minimizing production time, costs, as well as resources or raw materials. In this case, the actors of the production process must consider it as a common goal, and *collaborate* among them to achieve it, without neglecting their particular objectives, that is, they must deal with multiple objectives.

4. On the other hand, the smart factory can *cooperate* with other organizations in order to make automatic requests of raw materials in such a way to avoid stopping the production process. Finally, smart products can *cooperate* with other companies that offer courier services, so that the products reach the final consumer appropriately and in time.

In this Industry 4.0 scenario, the 3C processes are essential to achieve the goals of the production process.

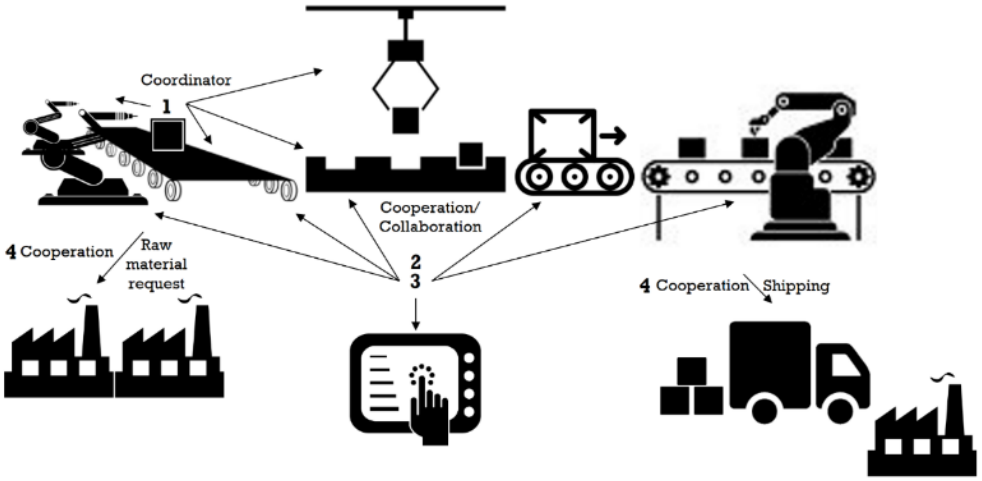

Figure 3.6. Industry 4.0 scenario with the 3C processes.

In order to promote coordination, cooperation, or collaboration processes, a CPS designed through MAS will be useful. In this sense, any actor in the production process will be characterized as an agent. Nevertheless, because the production process' actors have limited capabilities, many of their functionalities will be executed in the cloud, and eventually in the fog. It means that agents will incorporate the ability to establish communication with cloud and fog services. At this point, the integration architecture presented in the previous section plays a crucial role in order to allow agents and services to communicate. Figure 3.7 describes the communication process between the actors of the smart factory instantiated as agents, and the cloud and fog services.

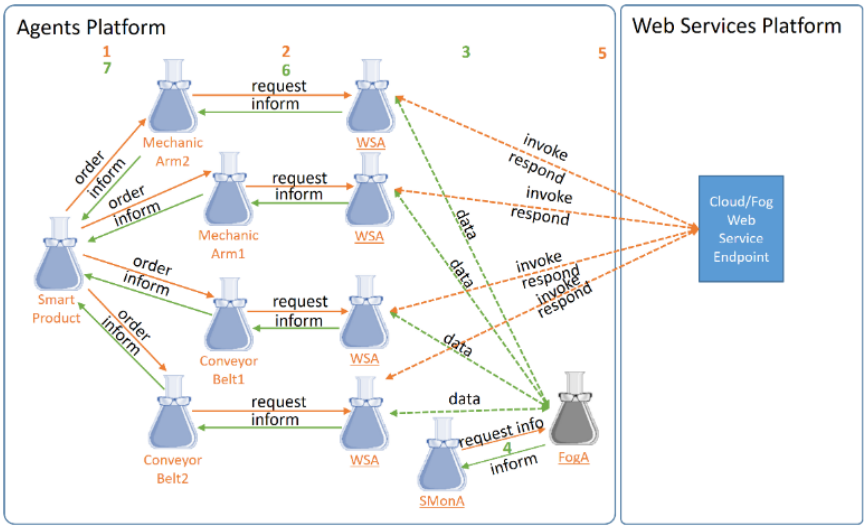

Figure 3.7. Instantiation of the Industry 4.0 case study using the MAS-SOA integration architecture.

Additionally, Figure 3.7 shows how the coordination process led by a smart product works.

1. The smart product gives the orders to other actors of the production system (arms, conveyor belt, etc.), according to the coordination plan that it handles.

2. These actors must perform specific tasks that allow adding layers to the final

product. When an actor requires a specific service, it must speak with the WSA agent that characterizes the corresponding service in the system (it is essential to remark that there is a WSA for each web service registered in the UDDI).

3. The WSA communicates with the FogA, to decide if the required service is located at the cloud or fog layers.

4. Then, the FogA requests the system information to the SMonA. Additionally, contextual information is retrieved using the ontologies component. This information helps to determine the needs of real-time processing, low latency, among other things. With this data, the FogA decides the location of the respective service and returns the result to the WSA.

5. WSA uses this information to adequately invoke the service (in the cloud or the fog).

6. Once that the WSA receives the results of the invoked service, it addresses that information to the requesting agent, so that it can finish his tasks.

7. Finally, the agent informs about the result of its task to the smart product agent, so that it can adequately adjust the coordination plan if it is needed.

8. We would proceed in a similar way to solve cooperation and collaboration cases.

In consequence, this research represents a significant contribution oriented to use MAS features in Industry 4.0, because it allows integrating actors of Industry 4.0 with the cloud computing paradigm, in order to deal with real-time needs, low latency, among other things, that are crucial in this context.

Notably, as it was explained before, all the devices on the production line are characterized as agents on the system, and they access services in the cloud or fog, to plan their tasks appropriately. For example, a smart product can access a process model as service, which has the knowledge base and specification on how to build the product. Also, a scheduling service can be useful in order to schedule the tasks of each device. On the other hand, device agents can request services that return the specifications on how to perform a specific task for a specific product (ex. trajectories, specific cuts, and 3d models), etc. In general, this contribution is significant to avoid real-time and low latency issues, oriented to incorporate the IA in production lines and to prevent stopping or delaying the production process.

3.6 Study case

The main goal of this chapter is to show that the proposed MAS-SOA architecture can solve the issues introduced by the cloud computing paradigm, autonomously and intelligently. In this sense, we need to test and verify that our solution is compatible with an agent-based system and that it provides better performances than other researches.

In this section, it is presented a case study that helps us to verify if the incorporation of the fog computing paradigm improves the performance and makes it possible to deal with real-time and latency problems, among other aspects.

Although theoretically, it is evident that the Fog Computing paradigm helps to solve the issues introduced by the cloud computing paradigm, in practice, the implementation of a solution as the proposed in the previous section, could not work, and by contrast, it might increase those issues. For this reason, it is imperative to verify the performance of the system in order to check if the incorporation of the fog computing component solves the existing issues of latency, real-time, among others. In this sense, three general case studies were designed. Moreover, one specific scenario was studied to compare our approach with previous works.

3.6.1 General Considerations for simulation.

The variables to consider in the simulation process are:

- Number of agents in the system that offer their functionalities as services.
- Number of web services registered in the UDDI. It means the number of services usable by agents.
- Maximum number of WSA agents supported by the platform. It depends on the system's resources.
- Average time of generation of new web service requirements by agents (Agent-> WS).
- Average time of generation of new requirements for agent services (WS-> Agent).
- Average response time of web services. This variable always remains constant since it affects all cases equally, and it does not influence the comparison of them.
- Standard deviation of the response time of web services. This variable always remains constant since it does not influence the performance of the proposal.
- Average response time of agents. This variable always remains constant since it does not influence the performance of the proposal.
- Standard deviation of the response time of the agents. This variable always remains constant since it does not influence the performance of the proposal.
- Average response time of requests for the SMA agent. This variable always remains constant since it does not influence the performance of the

proposal.

- Average time for message translation time (SOAP-ACL, ACL-SOAP). This variable always remains constant because it affects all cases equally, and it does not influence the comparison of them.

- Need for fog processing. It takes values between 0% and 100%, and it is only valid when the fog component is activated.

Each scenario presents a variation of these variables in order to study the specific system's characteristics. For all scenarios, a simulation time of 1800 seconds and a maximum of WSA agents of 2000 were set up. That is, the SMA platform cannot create more than 2000 WSA agents. In consequence, if there are many concurrent service invocations and it exceeds the maximum value of WSAs, then the next requests will be rejected until the WSA number drops again. In this way, we avoid oversaturating the system resources.

3.6.2 Simulation scenarios and results.

Case 1: In this case, the middleware is tested with low (10), middle (200), and high (1000) quantity of services. Additionally, it is evaluated the performance of the middleware with fog and without fog enabled. Besides, we used four levels of service requests: low level (10 requests each second), middle level (20 requests each second), high level (50 requests each second), very high level (100 requests each second). Moreover, in this case, it is assumed that there are low requirements of fog computing (need for real-time is low, and the Internet latency is very low too). In this sense, 20% of all generated requests will need real-time processing.

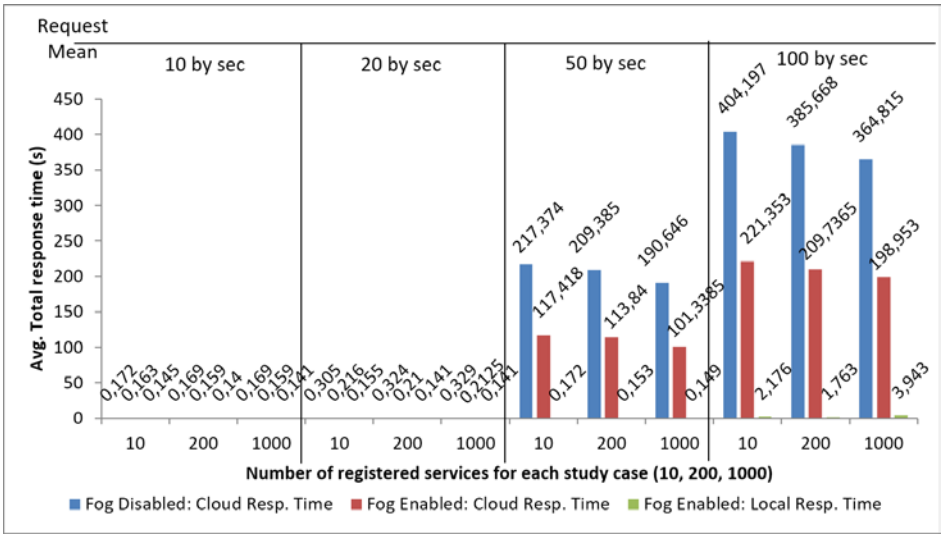

Figure 3.8. Average total response time vs. the number of services.

Figure 3.8 shows the average total response time (it includes lookup time of service, the invocation of the service, message translation, delivery time for the local message, and delivery time for cloud messages) against the number of services. The left bar indicates the average total response time for the request from the SMA to services in

the cloud when fog is disabled (it means that requests are processed in the cloud), the second bar corresponds to the total response time to consume services in the cloud when fog is enabled, and the last bar indicates the average total response time for requests with real-time needs (processed in the fog).

Additionally, it can be noticed in Figure 3.8, that when the mean of request is 10 or 20 requests by second, the average total response time is lower than 1 second in all cases. However, the average total response time for those requests that need real-time is always low. On the other hand, when the mean of request is 50 or 100 requests by second, the average total response time for those requests with fog enabled is always smaller than those requests where fog is not enabled.

In the same sense, and still more important, is that when fog is enabled, those requests that need real-time or low latency always have a total response time very close to 1s. Based on that, we can say that enabling fog computing in the original proposal, give us excellent results for those requests that need real-time or low-latency when the needs of real-time is low. This result is significant to enable real-time capabilities in applications in the context of Smart Cities, Cyber-Physical Systems, Industry 4.0, among others. For example, in a smart city, healthcare services like first aids, or vehicular information like traffic and geolocation, to users need real-time capabilities, and our solution solves this issue. In the Industry 4.0 context, this solution allows accessing production processes (BPaaS) or knowledge bases (KaaS), among other services, in the cloud.

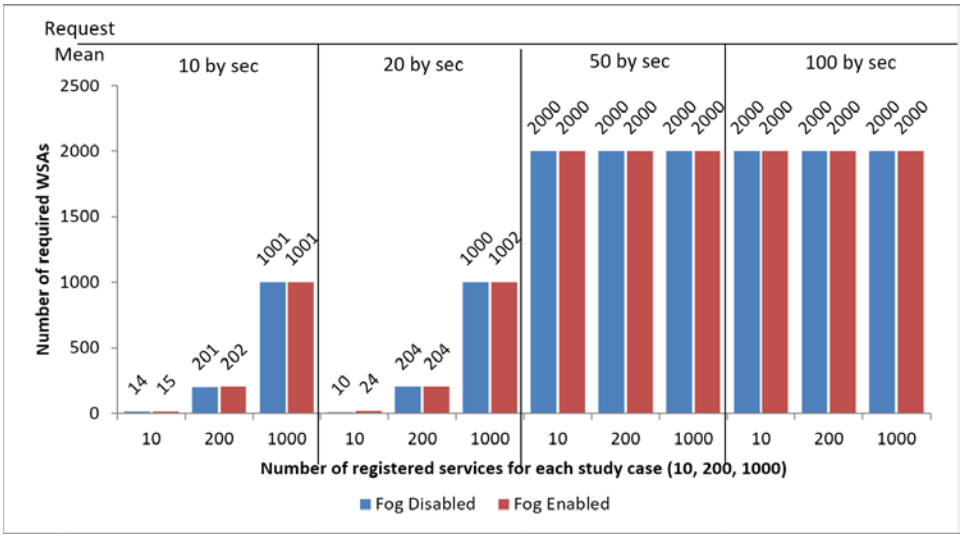

Figure 3.9. Number of required WSA vs. the number of registered services.

Figure 3.9 shows the number of WSA required to attend all the service requests generated on the system (left bar when fog is not enabled, the right bar when fog is enabled). According to Figure 3.9, whether or not fog is enabled, the system's behavior is similar. That means that the inclusion of the Fog Computing paradigm on the system does not change the deployment of the middleware for SOA-MAS integration.

In Figure 3.10, it is shown how communication of services in the cloud with the

agents in the platform (web services consuming agents as services) is affected when fog is enabled or disabled. It can be noticed that there is not a big difference between both results. It means that the inclusion of the Fog Layer on the system does not affect the communication in our SOA-MAS integration approach.

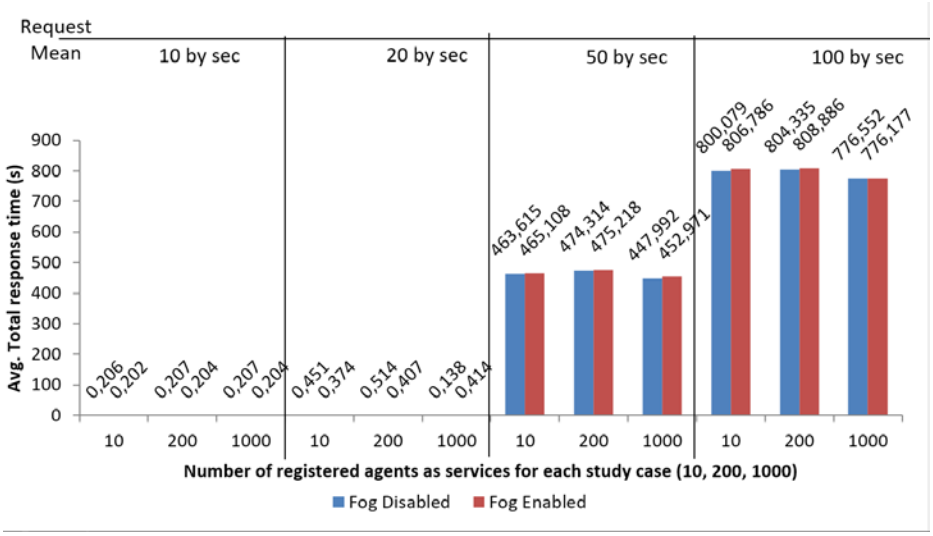

Figure 3.10. Average total response time vs. the number of registered agents as services.

Case 2: This case is an extension of Case 1, but it has high needs for real-time because it needs to solve a healthcare emergency that has arisen in the system. It means that 70% of services will require real-time processing.

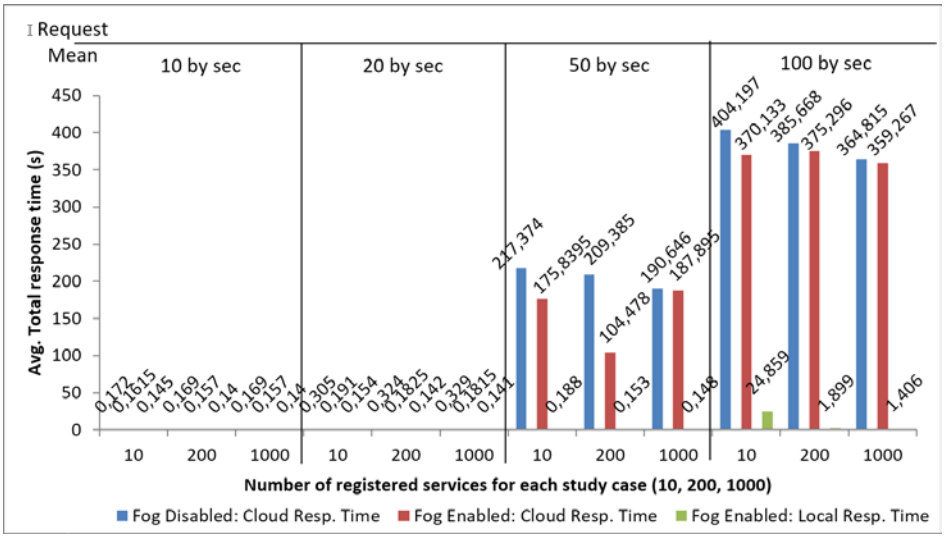

Figure 3.11. Average total response time vs. the number of services.

Figure 3.11 shows similar results, as in Figure 3.8. However, in Figure 3.11, it can be seen that, in general, all the requests that need real-time processing when the fog paradigm is enabled, were solved in lower time that those that require real-time, when fog computing is not enabled. Moreover, we can see a big difference in the average time when we have 50 or 100 requests by seconds. It also can be noticed, that when fog is enabled, those requests that are processed in the cloud take less time to be solved that when the fog is not enabled. It happens because the internet is less overfilled since

more requests are being processed in the fog. In general, this figure shows that when we have good internet speed and high requests of real-time, the system with fog enabled brings better results than when the fog is disabled, respect to the total average response time.

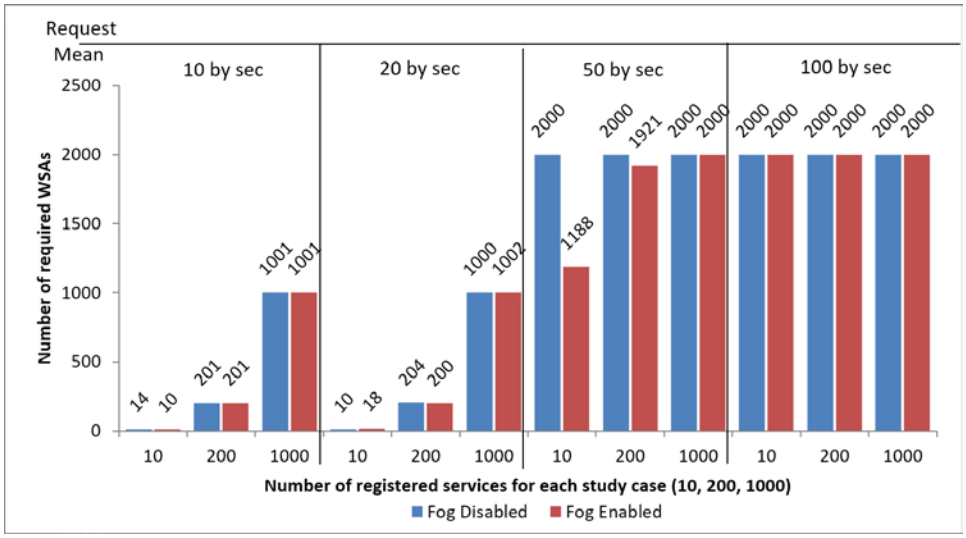

Figure 3.12. Number of required WSA vs. the number of registered services.

On the other hand, analyzing the system's requirements against the number of WSA needed to process all the requests generated in the system. Figure 3.12 shows that there is not a big difference when fog computing is enabled in the system, that when fog computing is disabled. Moreover, in Figure 3.13, it can be seen that the total average response time for SOA-MAS communication (services in cloud consuming agent tasks as services) is lower when fog is enabled for all cases. In that sense, we can say that enabling fog in the system, when internet connection is excellent and many services need real-time, allows reducing the response time in all cases.

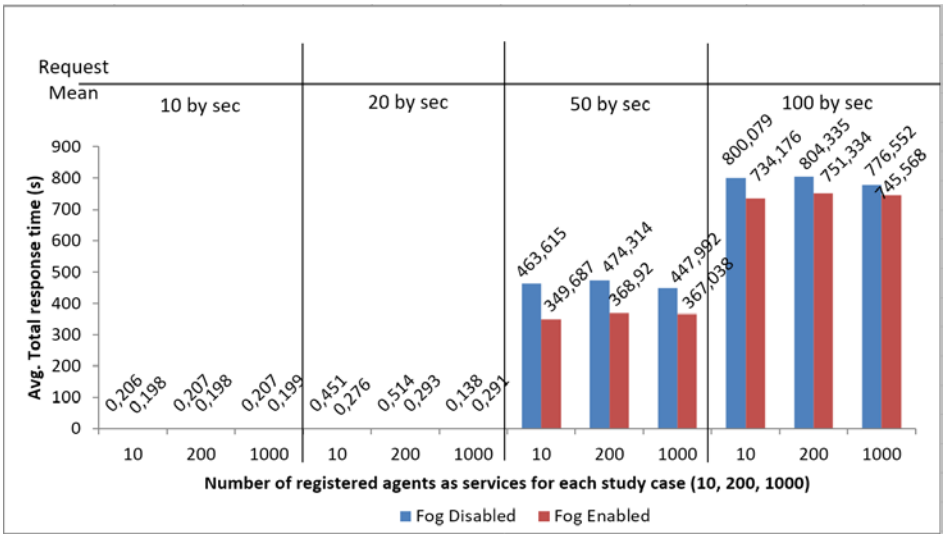

Figure 3.13. Average total response time vs. the number of registered agents as services.

Case 3: This case is also an extension of Case 1, but now, the internet's latency is

high (200ms), and we have real-time needs. That means that most services (80%) will require low latency or real-time processing.

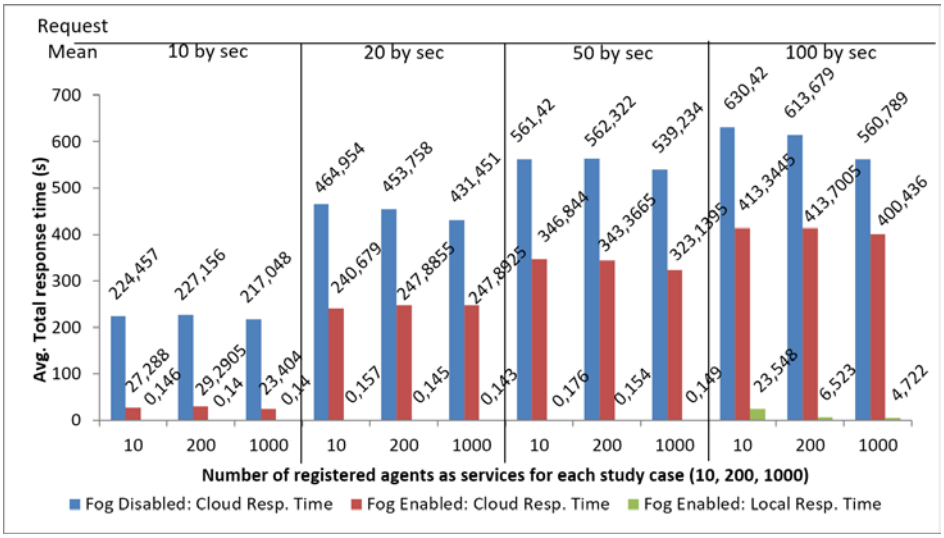

Figure 3.14. Average total response time vs. the number of services.

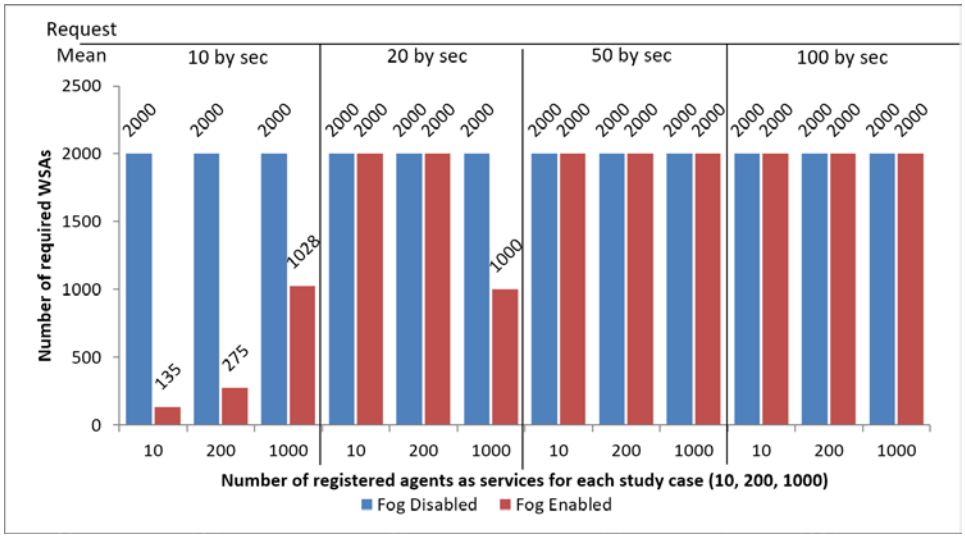

Figure 3.15. Number of required WSA vs. the number of registered services.

In this third case, we get similar results as in Cases 1 and 2. Mainly, from Figures 3.14, 3.15, and 3.16, it can be deduced that when fog is enabled in the platform, the total average response time (bidirectional), and the number of WSA agents required to process all the requests get best results. As a significant number of requests are processed locally, the system gets better results dealing with low latency for those requests processed in the cloud. It helps to reduce the total response time when fog is enabled. In contrast, when the fog is disabled, all data goes through the Internet, which increases the latency and increases the total response time in both ways of the communication channel.

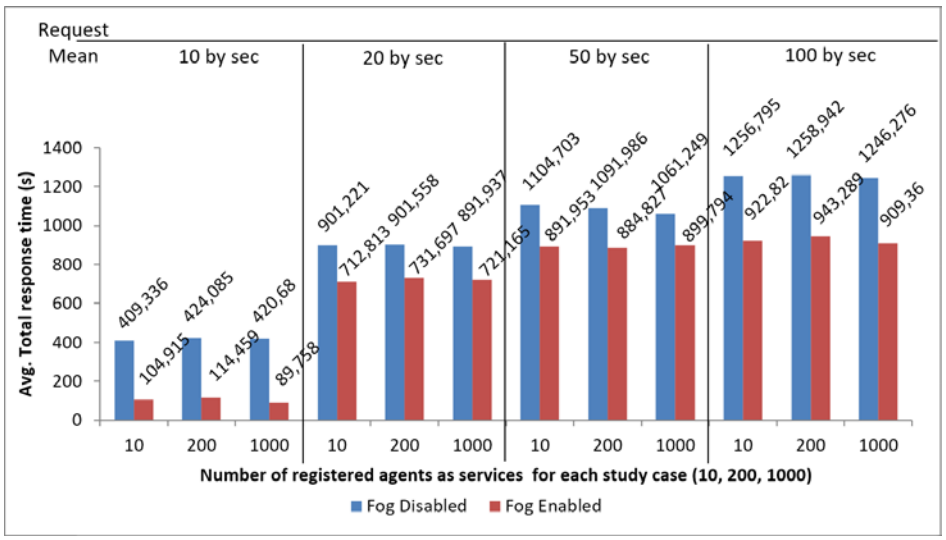

Figure 3.16. Average total response time vs. the number of registered agents as services.

Based on previous results, we can say that adding the Fog Layer to our proposal for the SOA-MAS integration, brings excellent results and allows the system to deal with low internet latency and real-time needs properly.

Case 4: This case was designed to allow the comparison of our proposal with other researches. Particularly, only Mohamed et al. (2017) presented quantitative metrics that can be used for comparison with our research work. Same as Mohamed et al. (2017), we used two web services, a very low latency of internet, and one web service request each second. In the same way, it was assumed that each service is invoked each second. We set up a simulation time of 60 seconds in order to fit the Mohamed et al. (2017) simulation. On the other hand, it was assumed that 50% of the requests need real-time processing. In this way, the case study proposed by Mohamed et al. (2017) was reproduced successfully.

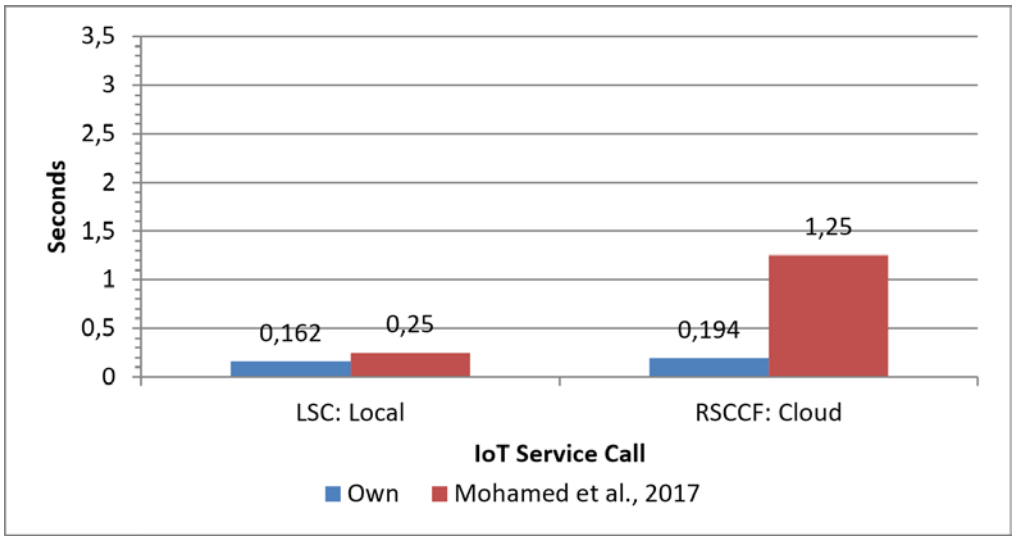

Figure 3.17. System (fog-enabled) response times (own: blue, (Mohamed et al., 2017): red)

Figure 3.17 presents the results of this simulation case. In this figure, LSC refers to services with real-time needs (processed locally), and RSCCF refers to services in the

cloud (do not require real-time), see (Mohamed et al., 2017) for more details. In our case, the response time for those services computed in the fog is 162 ms, while that the middleware proposed in (Mohamed et al., 2017) produced response times between 250ms and 500ms. On the other hand, the services processed in the cloud give us response times of 194 ms, while that in (Mohamed et al., 2017), the response time for services in the cloud was between 1250ms and 1500ms. Thus, our integration proposal gives better results that the middleware proposed in (Mohamed et al., 2017).

The difference between the response time of our middleware and the proposal of (Mohamed et al., 2017) is due to that our system combines semantic and context information from the system, in order to identify if the data received by the FogA require real-time processing. Our proposal uses the Fog Ontology (Aguilar et al., 2017e) to verify some properties, such as real-time needs that allow the FogA to decide the correct place of the service (local or cloud) in a lower time.

In brief, these simulations cases allow us to prove that the Fog paradigm was effectively integrated and that it causes a definite improvement in the quality of service, latency, among other things, concerning the version of this architecture without fog enabled. Also, regarding other research works, it proves that the interaction agents' diagram presented in Figures 3.1 and 3.5 is a suitable integration method for MAS-SOA-Fog paradigms because the performance of the middleware was improved in many senses. For instance, we can see from Figures 3.8, 3.11, 3.13, and 3.14 that the response time is always low, for situations that require real-time or low latency, and that it does not depend on the number of services registered on the system. Also, it evidences that the system's scalability against the number of services deployed in the system is guaranteed. Furthermore, Figures 3.9, 3.10, 3.12, 3.15, and 3.16 show that the platform requirements are not altered when fog is enabled. It means that our approach resolves all the cloud computing issues without losing performance or increasing its needs.

3.7 Results

In this section, we compare our approach with previous works. Table 3.1 resumes these differences using the following criteria:

1. SOA-MAS Enabled. It indicates if the research work allows communication between agents and web services.

2. Bidirectional SOA-MAS Integration. It indicates whether the agents can consume web services, and web services can invoke the agents' tasks.

3. SOA-MAS Enabled autonomously. It indicates whether or not the system administrator must create the gateway or communication channel manually to allow SOA-MAS integration, and if agents/web services need to make any message transformation. That means, all the communication is made transparently.

4. Fog Enabled. It indicates if the middleware supports the Fog computing paradigm.

5. Fog Enabled Autonomously. It indicates if the Fog Layer of the architecture is able to decide whether the data must be processed autonomously (cloud or fog), according to the current context and system performance.

Work	1	2	3	4	5
(Amadeo et al., 2017)	✓	×	×	✓	✓
(Peng et al., 2017)	✓	×	×	✓	×
(Giordano et al., 2016)	✓	×	×	✓	×
(Mohamed et al., 2017)	✓	×	×	✓	×
(Archimède et al., 2017)	✓	✓	×	×	×
(Al-Ayyoub et al., 2015)	✓	×	×	×	×
(García Coria et al., 2014)	✓	×	×	×	×
(Fuksa, 2014)	✓	×	×	×	×
(Pinto Pereira, 2014)	✓	✓	✓	×	×
Our proposal	✓	✓	✓	✓	✓

Table 3.1. Comparison with previous works

As it can be observed in Table 3.1, our MAS-SOA-Fog integration middleware is the only one that can autonomously integrate the SOA-MAS-Fog paradigms. It means that the agents and web services can communicate in a bidirectional way in order to solve the situations that arise in the environment, avoiding real-time and latency issues by autonomously using the fog computing paradigm. Besides, our SOA-MAS-Fog integration architecture can be used in different intelligent environment contexts. Mainly, this integration architecture was used to develop Smart Classrooms (Aguilar et al., 2017c, 2017d, 2015a, 2015b; Sanchez et al., 2015a, 2016), Smart cities solutions (Aguilar et al., 2016, 2017e), and others (Aguilar et al., 2017c; Sanchez et al., 2018b, 2015b).

3.8 Summary

SOA-MAS integration is a crucial point in any agent-based middleware in order to take advantage of the cloud paradigm. Such is the case of the Cyber-Physical Systems integrated with IoT, in which the cyber part can be associated with a MAS, and their service needs can be supplied in the cloud. Explicitly, the agents of the platform might use services in the cloud to process data (big data and data analytics services, linked data services, business process as a service, knowledge as a service, etc.), get recommendations, get context information, among other cloud services, facilitating intelligent decision making, context awareness, system recommendations, alerts, among other things.

Consequently, In this chapter, we have shown a component that allows the integration of MAS, SOA, and Fog computing paradigms in any intelligent environment, to allow agents and web services to communicate naturally, and in turn, to use the fog computing paradigm autonomously. The resulting solution allows taking advantage of the capabilities of both paradigms (MAS and cloud computing), while at the same time resolving typical problems of cloud computing-based systems, like low latency, real-time, geo-distribution, among others.

In particular, the integration middleware presented has shown excellent results in the case study. In that sense, it was proved that this solution provides exceptional results dealing with real-time problems and low latency issues. For example, from Figure 3.14, we can notice that when the Fog Layer is enabled, the average total response time is reduced considerably, having in most cases response times lower to 1 second, and in all cases, lower than 1 minute. That result is very desirable in real-time systems. In the same way, the average response time for services on the cloud is also lower when the Fog Layer is enabled because the traffic over the internet is low.

An essential remark since Figures 3.8, 3.11, and 3.14 is that the local response time is always low, and it has not linear dependencies against the number of requests. It means that our proposal guarantees low response time when the system requires real-time processing.

Regarding the proposal of Mohamed et al. (2017), it has been shown that our proposal provides better results in both cases, when invoking local services and when using services in the cloud. In general, the main difference of this research concerning other researches is that our work does not only allow MAS and SOA to communicate in a bidirectional way but also transparently and naturally. Thus, each agent does not need to worry about SOA specifications, and SOA services do not need to deal with FIPA details. On the other hand, other research works do not deal with real-time and latency issues. They only consider the integration of MAS and SOA paradigms, but, in most cases in a non-transparent way, and others only in one direction (from MAS to SOA), which means that SOA services cannot start a conversation with the agents (the system is not fully integrated). Our approach autonomously uses the fog computing paradigm according to the context and system requirements.

On the other hand, the case study presented in the context of Industry 4.0 shows that our integration proposal can easily be adapted to these systems. Such is the case of Cyber-Physical Systems, Smart factories, among others. Mainly, the result of this research can be useful in Industry 4.0, to integrate functionalities thought the use of the IoS paradigm (Sanchez et al., 2018a).

Chapter 4

Autonomic Cycles of Everything Mining for Coordination in the Context of the Industry 4.0

Contents

4.1 Introduction

After the implementation of the MAS-SOA-Fog integration middleware, we have decided to move forward in the 5C stack levels and provide solutions that not only supply the actors' communication needs but also to provide mechanisms for autonomous coordination cooperation and collaboration in the manufacturing system. This idea has arisen due that, as we discussed in previous chapters, the actors of

production processes (Things, Data, People and Services) should autonomously be able to act and make decisions, to implement self-* properties, such as self-configuration, self-management, and self-healing. In this chapter, we present a framework for the integration of autonomous processes based on the needs of coordination, cooperation, and collaboration of the production processes' actors. Notably, we define three autonomic cycles that allow the actors of manufacturing processes to interoperate autonomously. These autonomic cycles can create a coordinated plan for self-configuration, self-optimization, and self-healing during the manufacturing process. In this way, the actors are appropriately coordinated, such that they can autonomously manufacture Smart Products, detect failures, and recover from errors or failures, among other things.

4.2 Industry 4.0 and Coordination problem

Processes for coordination, cooperation, and collaboration (3C processes) are vital in manufacturing and industrial contexts because they can deal with issues such as integration and interoperability of actors, decision-making, and negotiation, among others. Moreover, the 3C processes will allow actors to negotiate and to achieve goals that cannot be accomplished by a unique actor. We consider that 3C processes are the central mechanisms for integration and interoperability in Industry 4.0. Essentially, these mechanisms will allow enhancing the autonomy of manufacturing processes. Furthermore, they can help to solve issues related to the heterogeneity of actors, distributed decision making, negotiation of production goals, among others.

Mainly, autonomous coordination represents a challenge that must be solved. In that sense, Haupert et al. (2017) propose the aggregation of semantic information to the data collected in a Smart Factory production line, to allow the creation of the service orchestration planning within a manufacturing process. Also, Syberfeldt, Danielsson, & Gustavsson (2017) say that a Smart Factory enables an extremely flexible production, and self-adaptable production processes, with machines and products that act both intelligently and autonomously. In that sense, the actors involved in the production line are empowered with services that allow interoperability among them. That means that the functionality of the manufacturing process is represented as a composition of different services. The authors affirm that the process can be easily improved by using a dynamic approach, which consists of adding semantic information to the services provided by devices. Likewise, the object (being manufactured) must provide a semantic description of how to produce itself. In this way, the orchestrator can use the semantic information attached to services, to create a dynamic orchestration of services, by adequately selecting the devices that will produce the object more efficiently. On the other hand, Haupert et al. (2017) test their proposal using the next metrics: a) how to endow a centralized orchestration process to an automated CPS production line; b) how a CPS can be self-maintained; c) how a service orchestration process can ensure that objects are produced efficiently.

Also, Leal et al. (2019) develop an ontology for interoperability assessment. The goal of this paper is to show which elements must be considered concerning the

interoperability assessment. The developed ontology is divided into two sub-ontologies: the assessment core and the systemic core. The assessment core permits the evaluation of the system's design. The systemic core describes concepts that enabled the design of different kinds of assessments. In the systematic core, the system is defined using characteristics of quality, which also can be related to the system's requirements. In counterpart, the assessment core contains concepts like the problem, the evaluation criteria, assessment scope, and assessment processes. Those concepts allow defining the reasons for what the assessment is made, as well as the quality attributes to consider, type of assessments, etc. The system interoperability is evaluated by defining the quality characteristics and the evaluation criteria, which must be rated to get a result that identifies the problems.

Lucas et al. (2018) use different communication technologies in a hierarchical architecture for communication and data management. The global operability of the system is guaranteed thanks to a central orchestrator. This orchestrator defines the data and communications protocols used by each sub-network, according to their requirements of latency and reliability. This work proposes grouping the devices into subnets or cells, which implement various technologies throughout the entire industrial plant. Each cell can use different communication technologies according to their needs. In this context, the orchestrator is responsible for the coordination of the resources assigned to different cells. The results show that this decentralized method can guarantee the delay necessities of the applications, and significantly outperforms a centralized approach (Lucas-Estañ et al., 2018).

A conceptual framework that allows achieving integrability, coordination and orchestration capabilities in a CPS was introduced by Rojas, Rauch, Vidoni, & Matt (2017). The framework is composed of five layers. The Control layer is an intermediate layer between the physical system and cyber units, which allows a proper integration of them. The Operational layer includes analytical tasks for monitoring, optimizing, and diagnosing the system, in order to empower the interoperability of the hardware in the control layer (The Operational layer is still under development). The Information layer is in charge of collecting data from all layers and provides high-level data analysis techniques in order to feed the knowledge bases. The Application layer is where APIs and User Interfaces are implemented. Finally, the Business layer refers to components of the upper layers of the automation pyramid (Hollender, 2010), like ERP and the decision making systems. In the framework proposed in Rojas et al. (2017), no autonomic coordination and orchestration mechanisms have been defined.

Cavalieri et al. (2019) proposes a solution for interoperability issues between OPC-UA and IoT (OPC-UA is a communication system based in a Publish/Subscribe pattern). The proposed solution uses the OCF (*Open Connectivity Foundation*) communication standard for information exchange on the IoT side. Furthermore, this solution allows an OPC-UA server to transfer all the information stored in it towards an OCF device, which translates the message and publishes it to other OCF devices in its ecosystem. Its architecture is straightforward but functional. However, the translation OCF to OPC-UA was not treated in that research work.

4.3 Industry 4.0 and Mining Tasks

Notably, this thesis project combines the Everything Mining, Autonomic Computing and the Internet of Everything (X. Chen et al., 2017; Martino et al., 2018; Shaikh et al., 2017; Yang et al., 2017) paradigms to solve the integrability and interoperability issues based on the 3C processes, in a manufacturing process. In that sense, we present a set of research works to describe how actors are currently using task mining to solve the Industry 4.0 issues.

Xu and Duan (2019) developed a survey related to the connection between CPS and big data in Industry 4.0. This study reveals that most researches are putting their efforts on the use of big data in the conception of CPS, but fewer researches focus on using the data analytics techniques to make CPS more efficient and effective. Furthermore, Xu and Duan affirms that most researches do not cope with the collaboration and cooperation of CPSs inter companies. Xu and Duan conclude by saying that using different data analytics applications will generate a high impact on the management of the whole Industry.

Consequently, Qin et al. (2016) proposes some mining tasks in the context of Industry 4.0. They have implemented a multi-layered framework of manufacturing for Industry 4.0. In the Integration layer, IoT (K. Chen et al., 2017; Mezghani et al., 2017a, 2017b) and CPS are used as technologies for the combination of the elements involved in the manufacturing process. On this level, sensors and machines are in charge of collecting the data produced in the supply chain, as well as to receive customers' feedback. Moreover, this level applies different data analysis technologies to discover useful information from data that can help to improve the manufacturing process. Also, technologies like Advanced Data Mining and Big Data Analysis are applied in the Intelligence layer to create a knowledge base that serves as a support for planning and decision-making processes. Notably, the Intelligence layer enables the manufacturing system to be self-aware, self-optimized, self-configured, and in general, self-*. The Automation layer is composed of physical components like machine and factories' processes. On this layer, technologies like PLC (Programmable logic controller), numeric controller, and statistical probability analysis, are used to optimize the production process.

Seeger et al. (2018) develop a solution that allows dealing with the scalability and performance issues generated in a system that has been configured dynamically by using IoT. For this purpose, a set of recipes are created. According to Seeger et al. (2018), a recipe is just a set of semantic descriptions of the configurations of the devices created in IoT. Moreover, a recipe describes the data flow between devices through ingredients that interact and exchange information. Likewise, these recipes allow the specification of restrictions that will impact the autonomous selection of the offer that best suits the instantiation of the recipe. Seeger et al. have verified that the scalability of the system is guaranteed due that the recipes are executed as a dynamic and distributed choreography rather than as a centralized orchestration process (Seeger et al., 2018). That means that choreographies are dynamically created according to the

system requirements. Also, the reliability of the system is guaranteed by a mechanism of failure detection and automatic recovery. In this case, when a device fails, it is removed, and a recipe is run in order to find a replacement.

In contrast, Wang, Wan, Zhang, Li, & Zhang (2016b) focuses on describing how MAS can be used in smart factories in order to allow autonomic coordination and cooperation processes. A negotiation mechanism leads agents to cooperate, allowing them to determine a route of agents to transport and elaborate the product. This negotiation mechanism is based on the contract net protocol, where the product acts as a manager. The MAS is supported by a Big data mechanism, which is used to solve the agent deadlocks and decision making.

4.4 Proposed Integration framework for Industry 4.0

This section presents the proposed architecture for autonomous integration and interoperability of actors in Industry 4.0.

4.4.1 Autonomic Integration Framework for the Industry 4.0 (AIFI 4.0)

In this research, we propose an autonomic integration framework for Industry 4.0 (see Figure 4.1).

This framework was conceived to use the Autonomic Computing Paradigm (Lalanda et al., 2013; Parashar & Hariri, 2005; Vizcarrondo et al., 2012), as an essential element that guarantees the autonomy and adaptability of the production process. Autonomic properties, like self-configuration, self-healing, self-optimizing, self-protecting, self-coordination (any self-* property) are developed in order to endows autonomy in manufacturing processes. Consequently, the *Physical layer* corresponds to the manufacturing process itself, where all the actors (People, Things, Data, Services) of the manufacturing process are involved, see section 2.3.4 for details about the actors:

In concordance with the Autonomic Computing paradigm, the Business Process is the Managed Resource, which means, it is the element that will be controlled in order to increase its autonomy. The Business Process is offered as a Service (BPaaS), which means that the Internet of Services (IoS) (Shila et al., 2017) is another essential paradigm integrated into this framework.

The Business Process resides in the Integration Platform as a Service for IoE component. This component is an intermediate point between actors, the Business Process itself, and the autonomic cycles in the Reflective Layer. Besides, it helps to solve issues related to the heterogeneity of actors and platforms. Consequently, the communication middleware presented in Chapter 3 will be plugged into the Integration Layer, depending on the technologies used to characterize the actors.

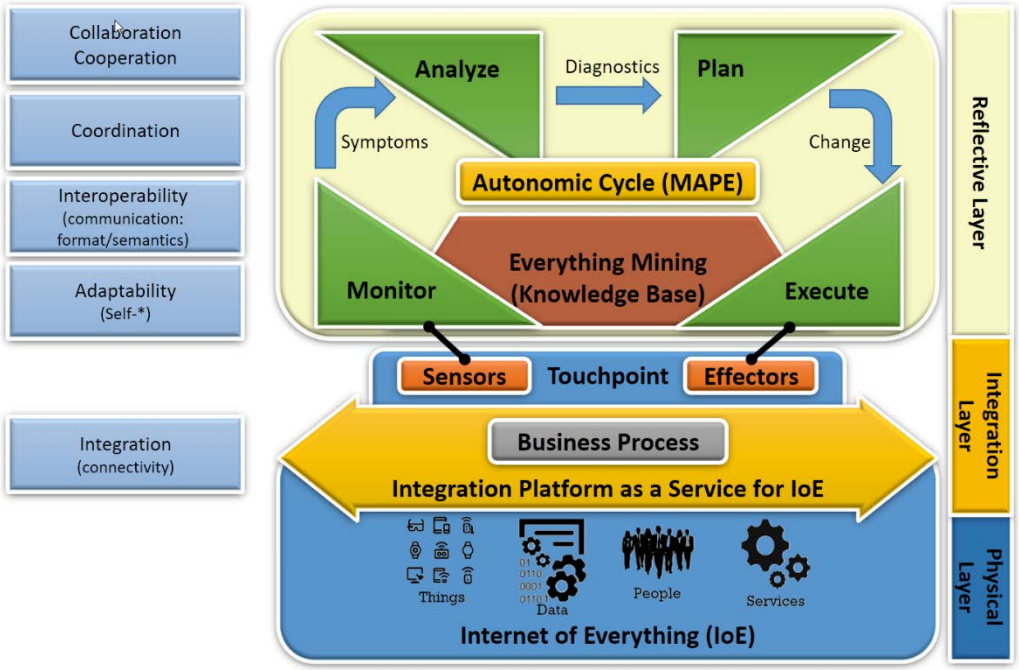

Figure 4.1. Autonomic Integration framework for Industry 4.0

Additionally, the Business Process is monitored continuously by the Reflective Layer in order to collect useful information for the creation of knowledge bases needed by the autonomic cycles. Moreover, in the *Integration Layer*, the communication among the actors involved in the Production Process is possible thanks to the IoS, especially, by the incorporation of the Internet of Everything (IoE), in a way that the actors are Things, Data, People, and Services. Typically, in other frameworks like the Cyber-Physical Production System (CPPS) (Qiu et al., 2017; Zanero, 2017), only Thing and Data are explicitly considered as actors of the production process. However, People and Services are quite crucial because they not only can address essential tasks (like decision-making, data-processing, and others), but they also produce valuable information about the production process that must be incorporated into the knowledge bases used for decision making. For instance, mining of the Business Process is essential in order to detect the production goals, the actors' metadata, the actors' task and roles, failures detection, bottlenecks detection, historical system performance, among others. It means that mining the Business Process can help in the development of self-supervising capabilities.

Expressly, this framework promotes the incorporation of autonomic coordination, cooperation, and collaboration processes as self-* properties to the Business Process *(Reflective Layer).* Consequently, the knowledge bases needed by those processes are created and improved by the Everything Mining component. This component is fundamental to allow our framework to learn from the Business Process and to encourage self-configuring, self-management, self-healing, and other self-* properties in the system. From this perspective, it is necessary to introduce the concept of "Everything Mining", as the mining of any actor, such as the process mining, big data mining, service mining, things mining, and people mining (sentiment analysis, opinion mining, etc.), with the primary goal of getting a better understanding of the system and

learn from it.

The information gathered by the Everything mining techniques will be used to create the knowledge bases needed by the autonomic cycles to make decisions. This knowledge is essential to promote autonomic coordination, cooperation, and collaboration in manufacturing processes.

In essence, the capabilities of the Autonomic Computing Paradigm based on "Everything Mining", such as the self-configuration, self-healing, self-optimization, and self-protection, endows autonomy to the system, making it more proactive, reflective and robust. For instance, the self-configuration property would allow the system to reconfigure itself according to its needs and objectives. On the other hand, self-optimization would help to improve the resources and raw materials utilization. Also, the self-healing property would allow detecting and repairing failures — for instance, the detection of delays in whatever stage of a production process. In the same sense, the self-protection characteristic would help to ensure system safety, for example, improving the security and privacy of the data.

From this example, it is easy to notice how the autonomy of the production process increases each time that a new self-* property is incorporated into the system. The fundamental feature of this framework is that it empowers autonomic integrability and interoperability of actors in manufacturing processes. This characteristic is possible thanks to the inclusion of self-coordination, self-cooperation, and self-collaboration (self-*) properties to the system.

Moreover, the proposed framework is based on the idea of adding autonomic cycles of Everything mining in a way that the self-* properties are added incrementally. For instance, we can start adding the self-coordination property. Next, we can add the self-supervising property, to finish with the self-reconfiguration property. In this way, the system's autonomy is improved incrementally in a simple way.

In particular, this chapter proposes three autonomic cycles with the primary goal of enabling autonomic coordination in production processes. The next subsections present these autonomic cycles.

4.4.2 Integration with RAMI 4.0.

In Chapter 1, was proposed an approach for solving the integration & interoperability issues incrementally in Industry 4.0, using the levels of connection, communication, coordination, cooperation, and collaboration (5C). This approach was called the 5C integration stack levels. The idea is to start solving the challenges at the connection level. Next, we pass to the communication level, finishing in the levels of coordination, cooperation, and collaboration, depending on the system's needs. In this section, we propose a framework for autonomous integration and interoperability of actors in Industry 4.0 called AIFI 4.0, which allows incrementally adding autonomous 5C processes as autonomic cycles. This architecture was described in the previous section. Moreover, in this section, we will show how this architecture follows other reference

architectures for Industry 4.0, as RAMI 4.0 (Platform Industry 4.0, 2018).

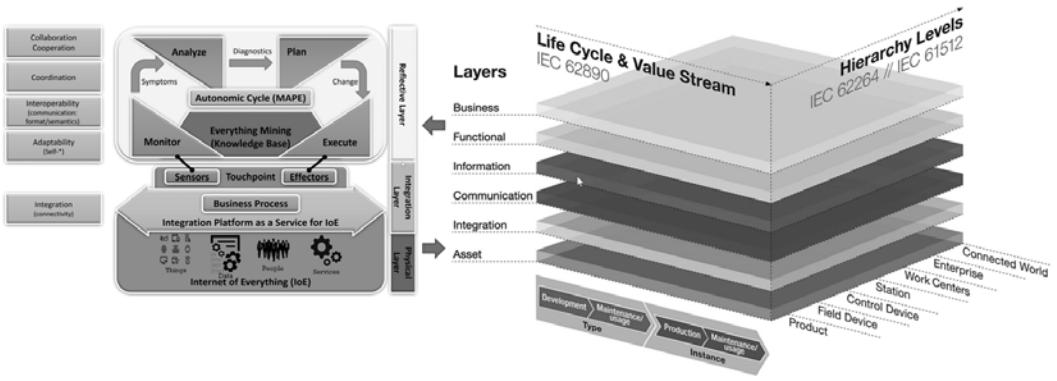

Figure 4.2. Compatibility AIFI 4.0 and RAMI 4.0

Figure 4.2 describes how AIFI 4.0 supports RAMI 4.0. Firstly, the physical layer of AIFI and the Asset layer of RAMI are the same. Secondly, the Integration Layer of AIFI is managed by the Internet of Everything. In this layer, all the physical elements are characterized as cyber components (digital twins). Besides, the Integration layer of AIFI brings support to the Integration and Communication layers of RAMI and lets actors to connect and communicate transparently. Thirdly, the Information Layer of RAMI has access to all the knowledge bases built using the everything mining paradigm. These knowledge bases ensure a proper understanding of the production environment. Fourthly, the Functional and Business Layers of RAMI are managed by the Reflective Layer of AIFI, which uses the MAPE loop to promote processes for self-coordination, self-cooperation, and self-collaboration of the asset, in order to achieve the production goals.

4.5 Autonomic Coordination in Industry 4.0

This section discusses the design of the AcoDAT in order to enable autonomic coordination in the manufacturing process.

4.5.1 Specification of the Autonomic Cycles for coordination in Industry 4.0

The Autonomic-manager for Coordination in Industry 4.0 (ACCI40) proposed in this chapter defines a set of data analysis tasks. The main goal of this autonomic manager is to allow self-planning, self-supervising, and self-configuring of the manufacturing process. In this context, the actors involved in the process can make decisions using the knowledge bases to improve the efficiency and productivity of the factory, detect failures, repair the system, among other things. Consequently, a Smart Product must be aware of guiding its production (that means, the Smart Product is smart enough to coordinate the actors to its manufacture).

In general, from our conception, an autonomic coordination process in Industry 4.0 requires a set of autonomic cycles of data analysis tasks, in order to create (self-configuring), supervise (self-supervising), and reconfigure (self-healing) the

production process. The autonomic cycles might use Everything Mining tasks to get useful information oriented to solve the coordination needs that arise in the production processes. In such sense, data and semantic mining tasks can be used to determine the objectives of the coordination process. In the same way, people and things mining tasks are useful to determine the elements that must be coordinated, as well as their availabilities, status, and roles. Furthermore, process and service mining tasks can help to determine failures in the production process, and contribute to self-healing, and self-optimizing of the production process. In conclusion, the concept of Everything Mining allows developing methods, tools, and strategies for the autonomic coordination in the production processes, within the context of Industry 4.0.

Notably, in this chapter, we propose a set of Autonomic Cycles of coordination tasks (called ACCI40), which can manage a production process. Specifically, the goals of each Autonomic Cycle are detailed as follows:

- ACCI40-1 (Build the coordination plan): This cycle is responsible for obtaining useful information for the creation of a coordination plan adjusted to the objectives and needs of the manufacturing process. The coordination plan should consider the availability of entities, their characteristics, roles, among others, in such a way that all the actors can be perfectly synchronized and coordinated. The outcome of this autonomic cycle is the prescriptive model of the coordination plan. A prescriptive model is a structured set of operations that describe how to achieve the objectives as efficiently as possible (Heldal et al., 2016). Moreover, prescriptive models are usually represented in a format that is readable by a machine or smart object.

- ACCI40-2 (Supervise the process): In this case, the tasks are oriented to supervise the coordinated manufacturing process, to detect failures, and to guarantee that the plan is being properly executed, among other things. Mainly, this autonomic cycle must analyze and predict actions, roles, along with others, more suitable tasks for the coordinated process. The input of this cycle is the prescriptive model of the coordination plan, and its outcome is the system's diagnostic model.

- ACCI40-3 (Self-configuration of the coordination plan): this autonomic cycle is responsible for the reconfiguration of the coordination plan when an abnormal situation is detected by the supervision cycle. In that sense, the coordination plan can be re-configured in order to solve the issues. The final solution considers the current manufacturing context (for example, if a device is out of service, then it must not be considered in the new plan, and its role will be reassigned to another device). Notably, this cycle generates a prescriptive model for the reconfiguration of the current coordinated process. The input of this cycle is the system's diagnostic model and the prescriptive model of the original coordination plan, and its outcome is a new prescriptive model for coordination, adjusted to the new needs.

The data sources for the previous autonomic cycles are the organization's databases (Inventory Systems, the Organization Social Networks, ERP systems, and the rest), the business process models, among others. Next, each autonomic cycles will be detailed.

4.5.2 Specification of the ACCI40-1: Build the coordination plan

The goal of this autonomic cycle is to build the production process' coordination plan, based on the production goals and the current context (information gathered from the process itself). Mainly, this cycle is composed of six tasks (see Figure 4.3):

- Determine the objectives of the coordination process.
- Determine the production tasks and the roles required.
- Determine the actors that must/can participate in the process.
- Determine the actors' activities based on their roles and competences.
- Determine the production requirements (technological or not).
- Design the coordination plan.

The first five tasks represent observation and analytics operations based on the process's data. In that sense, these tasks are in charge of extracting and analyzing the information about the production process to get useful manufacturing insights. Additionally, they must detect the needs for the production of a specific product, as well as the actors that must be involved in its production, the roles, and tasks to be carried out by each actor, among others. The ACCI40-1 is shown in Figure 4.3.

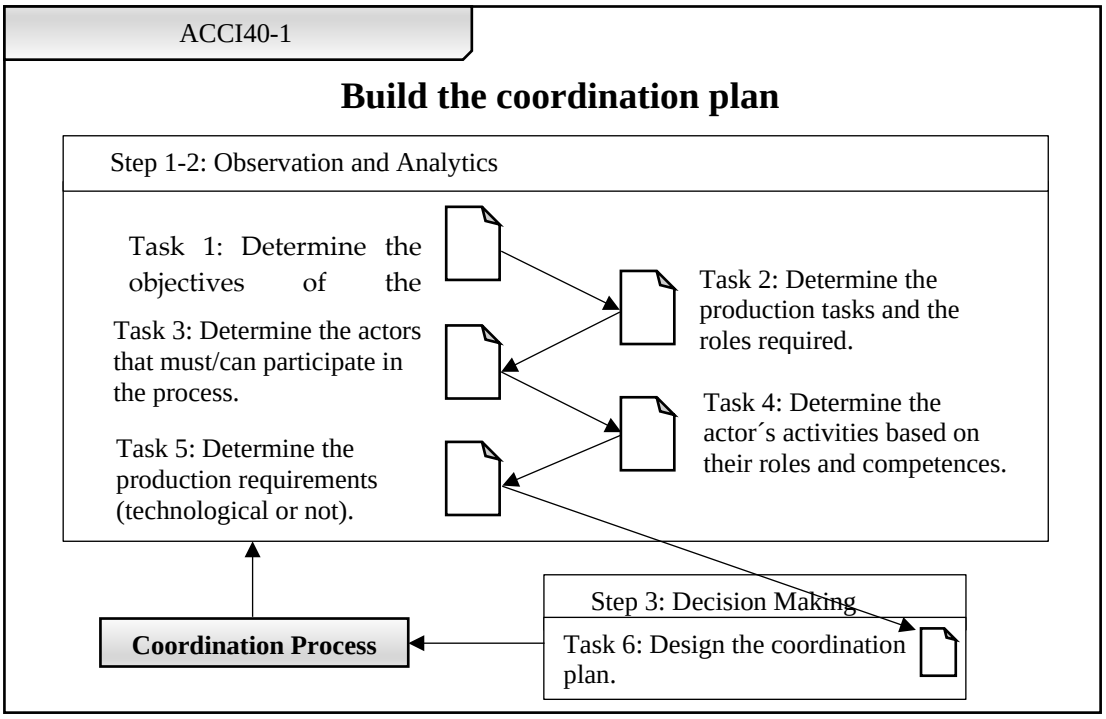

Figure 4.3. Structure of the ACCI40-1

Table 4.1 shows the data sources used in this autonomic cycle for each task. In the next subsections, the tasks for this autonomic cycle are described.

Task Name	Everything Mining Techniques	Data Sources
Determine the objectives of the coordination process	Process Mining, Data Mining	BPEL, Organization BD
Determine the production task and the roles required	Process Mining	BPEL
Determine the actors that must/can participate in the process	Thing Mining, People Mining, Service Mining, Data Mining	UPnP Services, BPEL Social network, SOA platform, Organization BD
Determine the activities for each actor, based on its roles and competences	Thing Mining, People Mining, Service Mining	UPnP Services, Bpel, Social network, SOA platform
Determine the requirements (technological or not)	Data Mining, Service Mining	Organization BD, SOA platform
Design the coordination plan	Data Mining	Previous Results

Table 4.1. Description of the Tasks of the ACCI40 1

Steps 1-2: Observation and Analysis Tasks.

Task 1. Determine the objectives of the coordination process: this task determines the specific production objectives using production data. For instance, it can determine new product requirements, detect if the products fit its current requirements, recognize if the production objectives have been met, among other things. For this purpose, the process, data, and semantic mining techniques can be used on data sources, such as social networks, organizational databases, business process models, etc.

Task 2. Determine the production tasks and the roles required: this task determines what production tasks should be carried out to manufacture the product, as well as the roles to be played by the actors of the process. In the same sense, this task must discover new and more effective ways to build products. Besides, this task will improve the mode of how the production tasks are assigned to each actor. For this purpose, process mining techniques may be used on business process models (BPEL, Petri nets, etc.) and event logs of the production processes, as data sources.

Task 3: Determine the actors that must/can participate in the process: this task looks for available actors to develop the production tasks. Additionally, this task can use predictive models to determine when the actors are available to synchronize and link to the production chain. In this case, it is necessary to use various mining techniques, such as thing mining, people mining, service mining, and data mining, on different data sources, such as the business process models, social networks, organizational databases, among others. For instance, it can use service mining over the SOA platform in order to infer the available services in the production environment.

Task 4: Determine the actors' activities based on their roles and competences. This task defines the assignment of production tasks to the process' actors, according to their availabilities, roles, and competencies. Similarly, several mining techniques are required, such as things, people, and services mining. Also, learning techniques can be useful to learn from the production processes about how actors must work more efficiently.

Task 5: Determine the production requirements (technological or not). The goal of this task is to analyze the technical aspects required by the production process in order to manufacture a product. For instance, it can predict whether the amount of raw material in the inventory is enough to produce the products in the current production order. This task performs data mining or service mining tasks on organizational databases and in business process models, among other mining techniques.

Step 3: Decision Making Tasks.

Task 6. Design the coordination plan: This task builds a prescriptive coordination model using the information obtained in the previous steps, in order to associate activities, actors, tasks, roles, the production process sequence, and the production requirements. That means that this plan describes how to assign the tasks to each actor based on the information collected from tasks 1 to 5, like roles, availability of actors, and other characteristics and requirements of the production process.

4.5.3 Specification of the ACCI40-2: Supervise the process.

The goal of this autonomic cycle, called ACCI40-2, is to supervise the execution of the previous plan, in order to ensure that it is working correctly. This autonomic cycle consists of 3 tasks:

- Determine how each actor is executing its tasks.
- Determine the general performance of the plan.
- Determine which actors guarantee the production process.

Now, we will describe the tasks of this cycle. Table 4.2 shows the tasks, data sources, and mining techniques of this cycle. The first task monitors the work of each actor in the process. The second task is an analysis task that determines the general performance and the status of the coordination plan. Finally, the third task is a decision-making task that determines the main actors of the production process. The structure of ACCI40-2 is shown in Figure 4.4.

Step 1: Observation Tasks.

Task 1. Determine how each actor is executing its tasks: The goal of this task is to monitor the production process, as a way of detecting how each actor is working (detect failures, delays, needs, task complexity, computational resources, etc.). This goal can be accomplished by applying things, people, processes, and service mining

techniques, among other techniques.

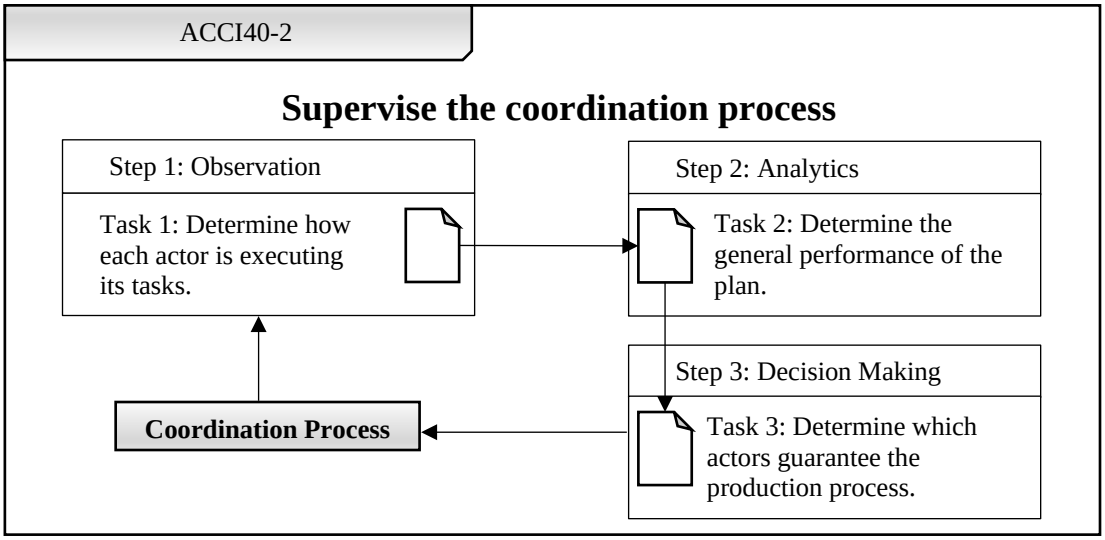

Figure 4.4. Structure of the ACCI40 2

Step 2: Analysis Tasks.

Task 2. Determine the general performance of the plan: This task is responsible for analyzing the general performance of the plan, based on the information provided by task 1, as well as by applying real-time process mining on the current production process.

Task Name	Everything Mining Techniques	Data Sources
Determine how each member of the teamwork	Thing Mining, People Mining, Service Mining	BPEL, Organization BD
Determine the general performance of the plan	Process Mining	BPEL
Determine which actors guarantee the production process	Data Mining, Thing Mining, People Mining, Service Mining	UPnP Services, Bpel, Social network, SOA platform, Organization BD

Table 4.2. Description of the Tasks of the ACCI40-2

Step 3: Decision Making Tasks.

Task 3. Determine which actors guarantee the production process: The main goal of this task is to discover, based on the information received from the previous tasks, the most reliable actors that guarantee to achieve the production goals successfully, with the lowest amount of failures or delays. Reliability, in this case, implies that the information received from these actors is valid and trusted. Reliability also means that the actors finish their tasks in the estimated time, among other things. Once this task has been completed, it generates a diagnostic model.

4.5.4 Specification of the ACCI40-3: Self-configuration of the plan.

The goal of ACCI40-3 is to use the information provided by the previous autonomic cycles, in order to detect issues and re-design the coordination plan according to the current needs. Notably, this cycle is composed of six tasks (see Figure 4.5):

- Determine the state of the coordination process.
- Determine the production tasks and roles required.
- Determine the actor's availability.
- Determine the actor's activities based on their roles and competences.
- Determine the production requirements (technological or not).
- Re-design the coordination plan.

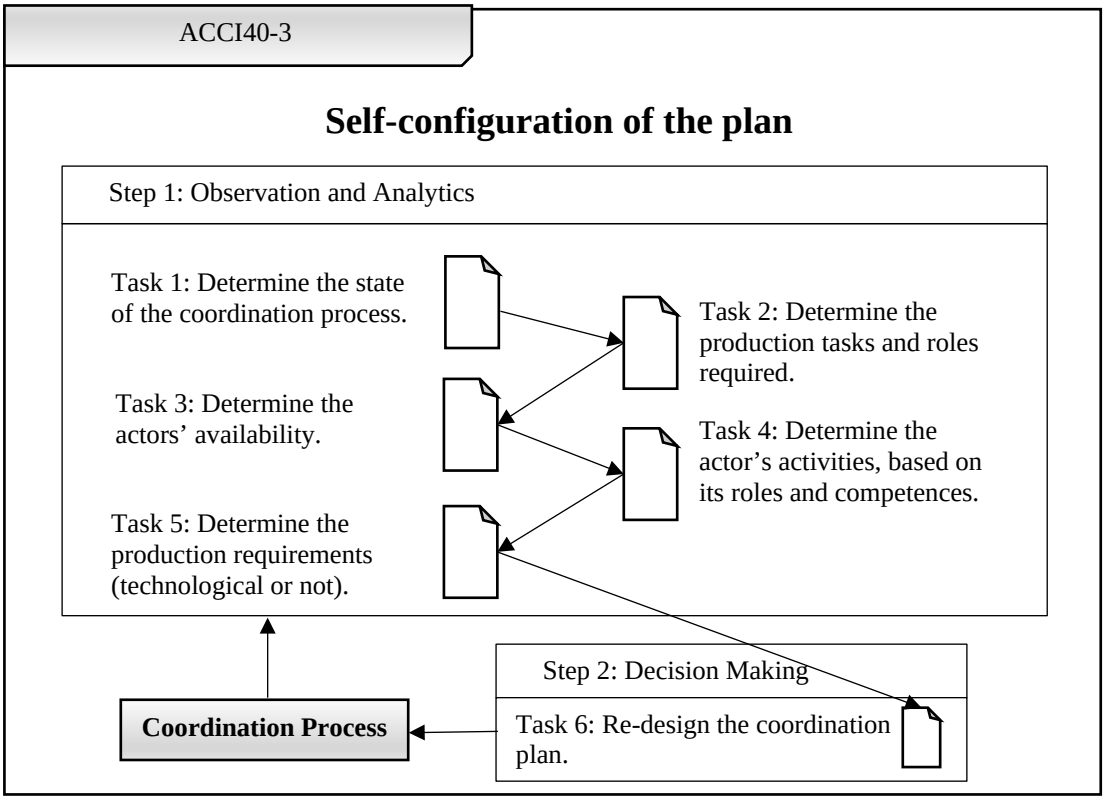

Figure 4.5. Structure of the ACCI40-3

Mainly, most of the tasks performed by this autonomic cycle are the same tasks performed by ACCI40-1. In this cycle, specifically, tasks 1 and 6 are different. Task 1 defines the current state of the coordination process, while task 6 designs the new coordination plan based on the current information provide by tasks 1-5. Table 4.3 shows the information about these tasks, as well as the data sources of the autonomic cycle. Only tasks 1 and 6 are described below.

Task Name	Some Everything Mining Techniques	Data Sources
Determine the state of the coordination process	Process Mining	BPEL
Re-design the coordination plan	Data Mining, Service Mining	Organization BD, SOA platform

Table 4.3. Description of the Tasks of the ACCI40 3

Step 1: Observation and Analysis.

Task 1. Determine the state of the coordination process: The main goal of this task is to observe the environment, in order to get information about the current state of the process (detect failures, delays, needs, task complexity, computational resources, among other aspects). The information collected by ACCI40-2 is essential for this task. Principally, the process mining techniques on the BPEL and event logs of the production process must be executed in order to achieve the goals of this task.

Step 2: Decision Making Tasks.

Task 6. Re-design the coordination plan: This task performs similar activities to task 6 of the ACCI40-1. However, it differs in the fact that this task is executed in real-time and must not introduce perturbations in the production process. Additionally, in ACCI40-1, the outcome is the prescriptive model that must be used for the coordinator to start the production process appropriately. In this case, the outcome is a new prescriptive model for coordination adjusted to the changes in the environment.

4.6 Case Study

4.6.1 Experimental Context

To illustrate the functionality of our autonomic cycles, we instantiated them into a generic production process (the factory is not smart), using the scenario presented in section 3.5. This generic scenario must be instantiated using a particular case study. This case study corresponds to a production process for the manufacturing of sandwich bread, where the client can customize the wrapper (logo, name, and other details), the quantity, the type of bread (white bread, grain bread, with raisins, with sesame, among others), among other things. The production process involves devices like the smart conveyor belt that route the bread to the least busy device, smart slicers that slice the bread, wrapping machine to pack the bread using the correct wrapper, the smart kneader machines, the smart printers, and the rest. Mostly, the production process is launched as a service using BPEL. Also, the organization has events logs and databases related to the production process, inventory, among others.

In this scenario, the coordination problem consists in how to produce the sandwich bread and customize orders for each customer (like logo, quantity, etc.), without increasing the production cost. Additionally, parameters like quality and resource consumption must be considered. Moreover, the selling price must be the same for big

orders as well as for small orders. This requirement can be achieved by grouping small orders between them, or by grouping small order with big orders. However, the coordinator must take care of how to separate them for distribution.

Formally, in this Case study, the Smart Product is the sandwich bread, and it is the coordinator of the production process. In that sense, the coordination process is based on autonomic cycles, according to the next steps:

1. The smart sandwich bread instantiates the ACCI40-1 to build the coordination plan. This plan contains information about devices selected for the manufacturing process, tasks assigned to each device, the sequence of operations that must be followed by the smart sandwich bread, the time for each device to accomplish its tasks, along with others. With this information, the smart sandwich bread can guide its production.

2. Once that the production process is started, the smart sandwich bread launches the second autonomic cycle (ACCI40-2), intending to start the self-management of the process and detecting failures during the manufacturing of the sandwich bread.

3. If some issues are detected by the ACCI40-2 (i.e., due to malfunction of devices, the detection of untrusted actors, etc.), the ACCI40-3 is started in order to self-heal the manufacturing process and continue with the production activities with the lowest delay.

The autonomic cycles use the knowledge bases created by the Everything Mining technique applied to transactional databases that contain information about orders, customer, inventory, recipes with information about how to prepare each type of bread, etc. as well as the event logs generated by previous executions of the manufacturing process, among others data sources.

According to the AIFI 4.0 architecture, all devices in the production line represent the physical layer. Moreover, for this case study, we are going to characterize each actor of the bread factory as cyber units using the MAS paradigm. The MAS will benefit from the communication architecture described in Chapter 3, in order to allow bidirectional communication between the cyber unit and the cloud computing services. Figure 4.6 describes the configuration that covers the two first levels of the 5C stacks. As can be seen, the elements of the bread factory are characterized by agents, an agent for each device. Those agents can control sensors and actuators of the device that they characterize.

Consequently, the multiagent platform uses the FIPA protocol for interoperation. Also, the agents in this platform can invoke cloud-computing thanks to the translator module incorporated in the WSA and WSOA agents. Besides, the FogA uses the Fog-Ontology, as well as the system information gathered by the SMonA oriented to remove the issues related to the cloud-computing based systems. IoE acts as a glue to allow the communication between all actors (services, people, things, and data). For instance, the business process is deployed as a service in the cloud; the recipe used to

prepare each type of bread is stored in the cloud. Moreover, human actors are integrated using an HMI. For instance, the Factory's Manager can connect from his office and change the production goals; the Factory's Baker can connect and change some conditions in the environment, etc.

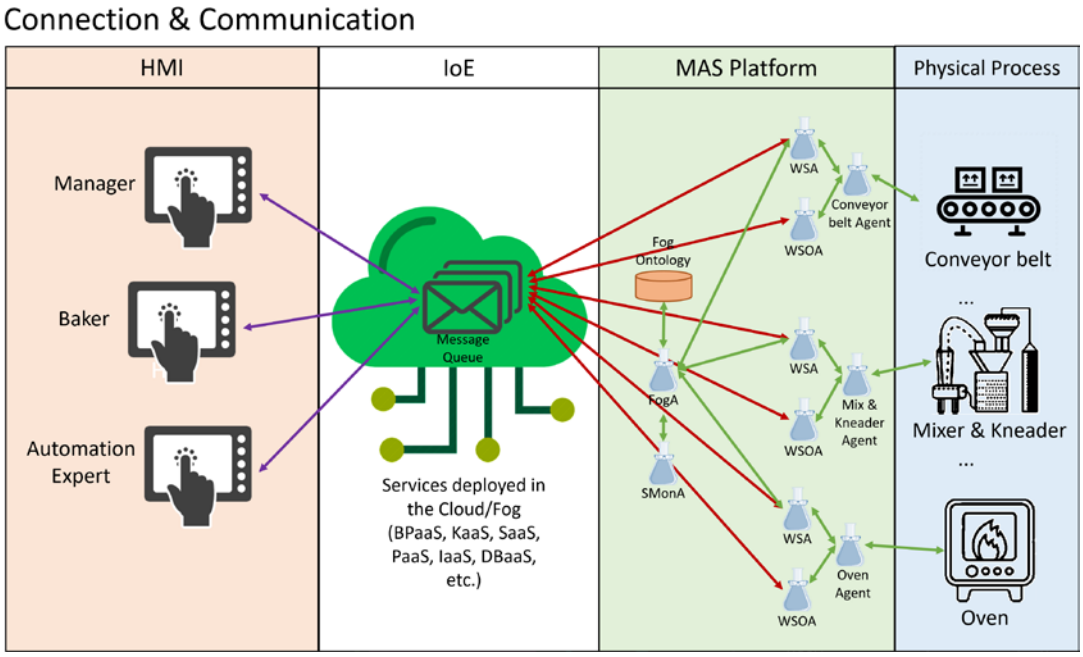

Figure 4.6. MAS Connection and Communication approach.

Notably, this configuration is beneficial to accomplish our goal of self-coordination due that agents can use for auto-organization, the autonomic cycle deployed in the upper layer of the AIFI architecture. It means that the coordination plan created by the self-coordination autonomic cycle can be used by agents to autonomously reach the production goals.

In the next subsections, the instantiation of the autonomic cycles is detailed. Those cycles are deployed in the reflective layer of the AIFI architecture.

4.6.2 Instantiation of the ACCI40-1: Build the coordination plan

At the beginning of the production process (before starting the manufacturing process), the Smart Product must configure the production environment, according to the current context. Moreover, it must consider the customizations requested by the Smart Product, as well as the availability of devices, among other things. The next steps describe how the ACCI40-1 is instantiated in this case study.

1. The Smart Product (smart sandwich bread) launches the first task of the autonomic cycle, in order to discover its production goals. It gets information about the production process (using process mining techniques) and customer orders (using data mining techniques). For instance, this step may discover that the production's objectives are to produce 1000 units of white bread, where 200 must be wrapped with logo A, 500 must be wrapped with logo B, and 300 with logo C, and 400 units of grain bread, where 200 must be wrapped with logo B,

and 200 with logo D.

2. The second task uses the information collected in the previous step in conjunction with the event logs, as well as the business process model in BPEL, in order to determine the activities that must be involved in the production process and the roles and production tasks that must be carried out by each actor. For instance, this process must return specific information like mix 15min, bake 45 min, stamp logo A, slice, package, transport, among others.

3. The Smart Product must determine the available actors. For instance, in our case study, it can discover that the smart conveyor belt must transport products through diverse phases of the production line and that only two slicers of three are available. Also, it determines if it is necessary to request authorization for one specific actor, among other things.

4. The previous information is used to define the criteria to assign activities correctly to each actor. Mainly, the gathered information in the previous tasks is used in order to associate roles and tasks with the actors that might be involved in the production process. Everything-mining techniques must be used as support to set the assignment. One example of the result of this task is: (smart conveyor belt, transport), (smart slicer 1, slice), (smart slicer 2, slice) (smart printer 1, print logo A), (smart printer 3, print logo B), (smart printer 3, print logo C), (manager, authorize shopping of raw material), etc.

5. Next, the Smart Product determines the production's requirements. For instance, this task defines the communication language and protocols between devices and other actors. One example of the result is: (smart printer, XML-SOAP), (smart slicer, JSON-REST), etc.

6. All the knowledge bases collected in the previous steps are combined to define the coordination plan, which includes information about the task assignment, actors' task sequence, among other things. For instance, the prescriptive model that represents the coordination is shown in Table 4.4. With that information, the Smart Product can start the production process. Also, as the devices are smart, they can make some decisions, and communicate with people (involved in the manufacturing process) when they need decisions that only the manager or another human actor could take, among other things.

The final result is shown in Table 4.4. It is a prescriptive model that indicates the tasks to be executed in order to achieve the production goals defined in task 1. Additionally, it contains information about the time required to finish each task, tasks' dependencies, as well as the tasks assigned to each actor. For instance, task number 3 (cut 1000 units of white bread) requires 30 min to be accomplished, should start after task 1, and might be executed by the Smart Cutter actor (see Table 4.4). The schedule plan can be generated using a Manufacturing Scheduling System (MSS) approach, as described by Rossie et al. (Rossit et al., 2019). The architecture of the automatic schedule generator is shown in Figure 4.7.

Nº	Task	Time (min)	Dependencies	Actor
1	Mix & Knead (1000 white bread)	45		Smart Kneader
2	Transport (Though Cutter)	30	1	Smart Conveyor Belt
3	Cut Bread (1000 white bread)	30	1	Smart Cutter
4	Transport (To Oven)	30	1	Smart Conveyor Belt
5	Bake (1000 white bread)	60	4	Smart Oven
6	Transport (thought slicer)	30	5	Smart Conveyor Belt
7	Slice (1000 white bread)	30	5	Smart Slicer 1; Smart Slicer 2
8	Transport (thought wrapping)	45	5	Smart Conveyor Belt
9	Wrap (1000 white bread)	45	5	Smart Wrapping
10	Transport (thought packer)	50	5	Smart Conveyor Belt
11	Pack (orders)	50	5;15	Smart Packer
12	Print (200 Logo A)	50		Smart Printer 1
13	Print (500 Logo B)	125		Smart Printer 2
14	Print (300 Logo C)	75		Smart Printer 3
15	Transport (to Packer)	5	12;13;14	Smart Conveyor Belt
16	Print (200 Logo B)	50	12	Smart Printer 1
17	Print (200 Logo D)	50	14	Smart Printer 3
18	Transport (to Packer)	5	16;17	Smart Conveyor Belt
19	Mix & Knead (400-grain bread)	30	1	Smart Kneader
20	Transport (Though Cutter)	20	19	Smart Conveyor Belt
21	Cut Bread (400-grain bread)	20	19	Smart Cutter
22	Transport (To Oven)	20	19	Smart Conveyor Belt
23	Bake (400-grain bread)	60	5;22	Smart Oven
24	Transport (thought slicer)	20	23	Smart Conveyor Belt
25	Slice (400-grain bread)	20	23	Smart Slicer 1; Smart Slicer 2
26	Transport (thought wrapping)	30	23	Smart Conveyor Belt
27	Wrap (400-grain bread)	30	23	Smart Wrapping
28	Transport (thought packer)	35	23	Smart Conveyor Belt
29	Pack (orders)	35	18;23	Smart Packer

Table 4.4. Coordination Plan generated by ACCI40-1

The everything mining layer, in Figure 4.7, represents the tasks (1), (2), (3), and (5) of this autonomic cycle. This layer collects the information needed by the schedule generator to create the plan (this information is described in the steps above). Besides, this layer feeds the knowledge base layer. The automatic schedule generator layer uses the information stored in the knowledge bases, and apply the MSS approach, to generate the plan, as described in tasks (4) and (6). This plan is sent to the Schedule Validator layer, in which an expert checks, adjusts, and approves the plan. The schedule validation process can be automated to check the plan, and if it is not approved, then it resents to the Automatic Schedule Generator in order to make the corresponding fixes.

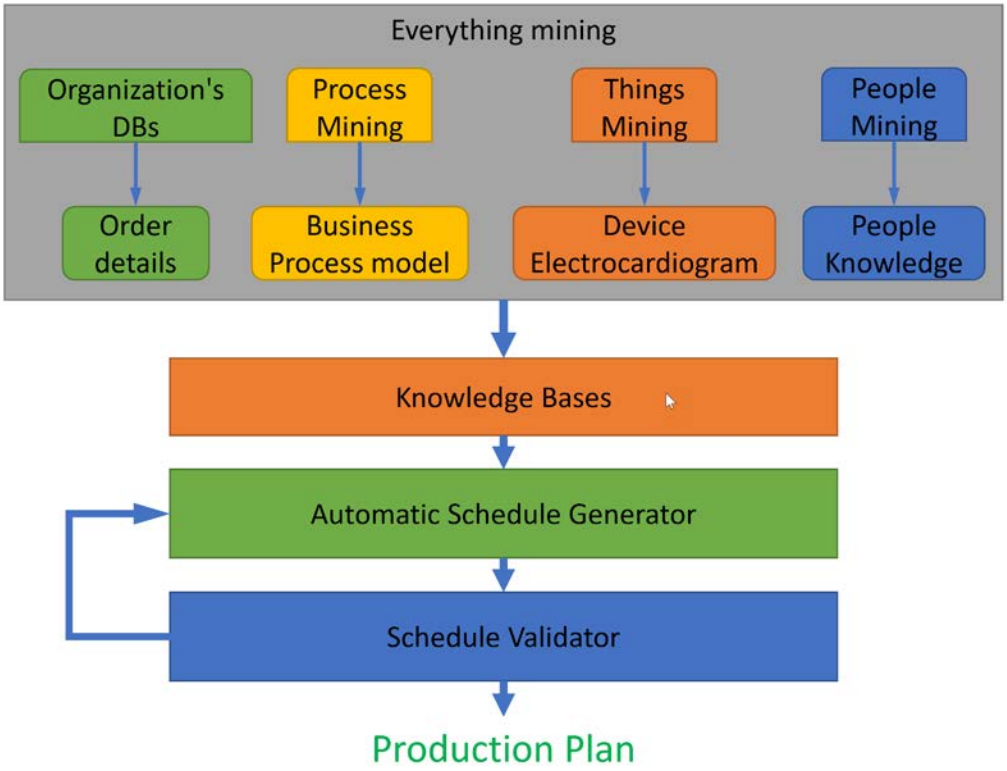

Figure 4.7. Automatic Schedule generator (Rossit et al., 2019).

4.6.3 Instantiation of the ACCI40-2: Supervise the coordination process

Once the process has started, the Smart Product must control the correct execution of the process, detect failures, among other things. In this sense, it launches the autonomic cycle ACCI40-2, in order to allow self-supervising. Notably, the steps followed by this autonomic cycle are:

1. The first step collects information from the actors of the production process using Everything Mining tasks. They reveal useful information to determine how each actor is working. Remarkably, it can determine what actors are delaying the process, what is the cause of the delay, among other things. For instance, it can discover that Smart printer 3 is causing a delay because it took more time to print the logos than other printers. In this step, the process mining technique applied to historical event logs could be used to create a knowledge model that contains information about the time used by each machine to accomplish their tasks.

2. In this step, process mining techniques are applied in order to measure the global performance of the production process. Mainly, this task compares previous executions of the production process with the current one and detects differences in the execution time. For instance, it can detect that although all the actors are working well, the production process presents a delay concerning its previous executions. Similar to the previous step, the model created using the process mining contains information about the global time employed to produce each order. This information is compared to the current execution, in order to

determine if the performance of the current process is delayed respect to the historical production data.

3. The third step is to detect if actors are executing their tasks correctly. Additionally, it can detect if the data sent by actors is trusted, or if there are communication failures, using everything mining techniques. For example, this step could use a predictive model built using historical production data, in order to detect if given the current conditions a product pass or fails the quality control test. One example of the output of this task is: (Slice 1, trusted), (Slice 2, trusted), (Printer 3, untrusted), etc.

Each time that a new knowledge model is added to the self-supervising autonomic cycle, the system gains more autonomy, and more failures can be detected. Table 4.5 shows an example of the diagnostic model generated for this cycle. It shows that almost all actors are working without issues, but the smart packer is generating some delays in the production process, and Smart Printer 3 presents a failure. That means, the production process requires to be reconfigured.

4.6.4 Instantiation of the ACCI40-3: Self-configuration of the plan.

This cycle takes the diagnostic model produced by ACCI40-2 as input, in order to decide about starting or not the reconfiguration of the manufacturing process. ACCI40-1 and ACCI40-3 share most of their tasks. Particularly in ACCI40-3, we are going to obtain a new plan similar to the one showed in Table 4.4, but containing new actors, new timing, among other things, depending on the current needs and context of the production process.

Also, it generates a new prescriptive model in order to improve the production process. This prescriptive model contains useful information to reconfigure the system. This model deletes actors that cause delays, failures, etc. in the process. Moreover, synchronization times are adjusted according to the time that actors currently take to execute each task. Thus, smart factories can gain autonomy in their collaboration processes as well as solve integration and heterogeneity issues, by including the cycles of data analysis tasks described in this section.

Actor	Status	Trusted
Smart Kneader	Alive	true
Smart Conveyor Belt	Alive	true
Smart Cutter	Alive	true
Smart Oven	Alive	true
Smart Slicer 1	Alive	true
Smart Slicer 2	Alive	false
Smart Wrapping	Alive	true
Smart Packer	Delayed	true
Smart Printer 1	Alive	true
Smart Printer 2	Alive	true
Smart Printer 3	Failure	true

Table 4.5. Diagnostic Model generated by ACCI40-2

4.7 Results

In this section, firstly, we present some premises intending to make a comparison with previous works and to show the advantages of using our proposal.

4.7.1 General premises for the integration of actors in the context of Industry 4.0

We propose the next premises, in order to determine the capabilities of our autonomic cycle for coordination in Industry 4.0:

- *First Premise*: Actors belonging to the production process attain their vertical and horizontal integrations by coordinating their interactions.
- *Second Premise*: Actors of the production process (data, people, things, and services) can communicate among them (interoperate), by sending and receiving data during the coordination process. This premise checks the communication level.
- *Third Premise*: the coordination is managed by the autonomic cycles of data analysis tasks.
- *Fourth Premise:* All actors involved in the manufacturing process should work together (interoperate), each one performing its specific tasks. This premise intends to check the independence of actors.

Regarding the first premise, our proposal contains different integration types thanks to the *Integration* and *Reflective* Layers of this framework. Vertical integration is reached by the autonomic coordination of actors within the same industry, while that the horizontal integration is reached by the autonomic cooperation and collaboration processes that take place between industries (see Figure 3.6). In our case study, vertical integration allows manufacturing the smart bread, thanks to the coordination process discovered by the ACCI40-1. In the same way, horizontal integration allows this industry to cooperate/collaborate with other companies in order to deliver orders, get raw materials, etc.

For the second premise, the interaction of actors is guaranteed due to the Internet of Everything paradigm. Moreover, The People actor can interact with other actors using the HMI devices disposed within the production process. In general, the framework allows the exchange of data between actors and their understanding, which guarantees that they can communicate (see Figure 4.1).

Concerning the third premise, the Smart Product invokes the ACCI40-1 in order to coordinate all the activities that need to be developed by other actors involved in the manufacturing process. In the case study, the smart sandwich bread invokes the ACCI40-1 to start the coordination process. Next, when the production process begins, it launches the ACCI40-2; and when a failure arises in the production process, the ACCI40-3 is invoked.

Finally, for the fourth premise, the coordination plan defined by the ACCI40-1 assures that all actors work together, in order to achieve the production goal. Moreover, the ACCI40-2 and ACCI40-3 allow reconfiguring the manufacturing process when a fault or failure is detected. They ensure the continuity of the production process. In our case study, the smart sandwich bread controls its production, and the autonomic cycles gather all the information that is needed to give orders to other actors. This premise guarantees the completion of the manufacturing process successfully.

4.8 Comparison with previous works

In this sub-section, a qualitative comparison with related works is made. In the first place, we verify if related works comply with the four premises defined in the previous sub-section (see Table 4.6).

As can be noticed from Table 4.6, almost all the previous works comply with all premises, except (Haupert et al., 2017), (Qin et al., 2016) and (Wang et al., 2016b). Substantially, Haupert et al. (2017) and Wang et al. (2016b) do not comply with the first premise because they only allow vertical integration. On the other hand, the works (Haupert et al., 2017) and (Qin et al., 2016) do not act following the third premise, because they do not use data analytical tasks.

	First Premise	Second Premise	Third Premise	Fourth Premise
(Haupert et al., 2017)	×	✓	×	✓
(Lucas-Estañ et al., 2018)	✓	✓	✓	✓
(Rojas et al., 2017)	✓	✓	✓	✓
(Qin et al., 2016)	✓	✓	×	✓
(Seeger et al., 2018)	✓	✓	✓	✓
(Wang et al., 2016b)	×	✓	✓	✓
This work	✓	✓	✓	✓

Table 4.6. Comparison with previous works based on the premises

Now, it is presented a qualitative comparison based on some specific characteristics that indicate the grade of autonomy reached by our framework. The next characteristics are used for this comparison:

1. Integration of the four actors of the production process.
2. Support Everything Mining.
3. Support the self-configuration of the production process.

4. Support the self-optimization of the production process.
5. Support the self-healing of the production process.
6. Support processes of self-coordination, self-cooperation, and self-collaboration.
7. Autonomic properties can be added incrementally.

Table 4.7 shows the result of this comparison. As can be seen, most of the related works do not satisfy all characteristics.

	1	**2**	**3**	**4**	**5**	**6**	7
(Haupert et al., 2017)	×	×	✓	✓	×	×	×
(Lucas-Estañ et al., 2018)	×	×	×	×	×	×	×
(Rojas et al., 2017)	×	×	×	×	×	×	×
(Qin et al., 2016)	×	×	✓	✓	×	×	×
(Seeger et al., 2018)	×	×	✓	✓	✓	×	×
(Wang et al., 2016b)	×	×	×	×	×	×	×
This work	✓	✓	✓	✓	✓	✓	✓

Table 4.7. Comparison with previous works based on the characteristics that indicate the grade of autonomy reached

Specifically, the works (Haupert et al., 2017; Lucas-Estañ et al., 2018; Qin et al., 2016; Rojas et al., 2017; Seeger et al., 2018; Wang et al., 2016b) do not allow integration of the four actors of the production process (1), because most of them use IoT as an integration framework what only consider the Things actor. That fact is the first weak point in those works because when services, people, data, and things are integrated, the framework can get a better comprehension of the production process, and the autonomy of the whole system can be increased. Additionally, (Haupert et al., 2017; Lucas-Estañ et al., 2018; Qin et al., 2016; Rojas et al., 2017; Seeger et al., 2018; Wang et al., 2016b) do not support everything mining (2). This element is a second weak point of those works because the data component is not the unique actor that can bring information to the process. With Everything Mining, we can get information from social networks (people mining), sentiment analysis (people mining), process, and service mining, device electrocardiogram of things, among other sources, which allows us to have a better understanding of the context. For instance, people mining can retrieve information about the conditions in which people produce the most. Also, the process mining could get information about how a product is produced (tasks of each actor, actors required in the production process, competences, among others).

Regarding the self-configuration of the production process (3), only works (Lucas-Estañ et al., 2018; Rojas et al., 2017; Wang et al., 2016b) do not comply with it. These

works do not allow the system to be self-configured, a characteristic that is essential in the Industry 4.0 context (Bahrin et al., 2016; Qin et al., 2016; L. Xu et al., 2014). Additionally, (Lucas-Estañ et al., 2018; Rojas et al., 2017; Wang et al., 2016b) do not support self-optimization of the production process (4), in that sense, they are not able to supervise the process to detect tasks that could be optimized, and launch the self-configuration autonomic cycle to repair the system.

Works (Haupert et al., 2017; Lucas-Estañ et al., 2018; Qin et al., 2016; Rojas et al., 2017; Wang et al., 2016b) do not support self-healing of the production process (5), which means, they are not able to supervise the process to detect failures and reconfigure it when an error occurs in the system.

Finally, any of the previous research works support self-coordination, self-cooperation, and self-collaboration (6). In particular, (Haupert et al., 2017) supports self-planning, but it is not based on coordination, cooperation, and collaboration processes, and neither applies to process and services mining tasks. They only propose a sequence of steps to carry out. The autonomy of the works presented in Section II is not good enough to let actors of the production process manufacture a product according to the requirements of Industry 4.0. Finally, the proposed framework in this work adds incrementally self-* capabilities, which means that the self-configuration can be added first. Next, the self-supervising capability, and so for the rest. Moreover, the other 3C functions can be posteriorly added, like the self-cooperation and self-collaboration, which will highly increase the autonomy of the system. In this sense, the scalability of the framework is guaranteed. This property is not found in previous research works.

4.9 Summary

In this chapter, we are focused on providing a solution for the integrability and interoperability needs in production processes regarding the context of Industry 4.0. The proposed framework can deal with the heterogeneity of actors and integration issues, thanks to the adoption of the Internet of Everything as integration Layer, as well as the use of the "autonomic cycle" concept. While the autonomic cycles can create/discover a production plan, in order to allow actors to interoperate, the Internet of Everything serves as the glue that joins all the actors together. That means it allows them to connect, communicate, and exchange data in order to execute their tasks.

Additionally, the Autonomic computing paradigm brings autonomy to the system, supported by the knowledge created by the Everything Mining component, which is in charge of the application of a variety of mining techniques, like data mining, sentiment analysis, social network analysis, service mining, process mining, among others, to determine useful information that might improve the system integration and interoperability. In that sense, the autonomic cycles represent an essential characteristic of this framework, in the first place, because they allow the autonomic configuration of the production process, enabling actors to interoperate and work

coordinately, in order to produce Smart Products. Additionally, the autonomic cycles apply an intelligent loop that supervises the whole system, in order to know whether or not the actors are working correctly and to reconfigure the process autonomously when any issue is detected.

Mainly, the autonomic cycles for coordination presented in this work, use the information collected by the Everything Mining tasks, to autonomously create/discover, supervising, and optimizing/repairing the production process. Also, the autonomic cycles allow our proposal to increase the autonomy of the manufacturing process, which is an essential feature in the Industry 4.0 context (Bahrin et al., 2016; Lu, 2017; Qin et al., 2016; L. Xu et al., 2014).

The next chapter shows how these three autonomic cycles can be implemented using a real case study.

Chapter 5

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

Contents

5.1 Introduction

The vision of 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. The previous chapter proposes three autonomic cycles of data analytics tasks for self-coordination in manufacturing processes. In this chapter, we implement one of these autonomic cycles in order to allow self-supervising of the coordination process. This autonomic cycle for self-supervising 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.

Essentially, 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). This analysis produces predictive data that is used to gain the capabilities of self-awareness and self-maintenance. These capabilities contribute considerably to the resilience, automation, and productivity of manufacturing processes in order to make predictive decisions about machinery failures and machinery deterioration trends. Xu et al. (2017) establishes the importance of having the right diagnostic approach to guarantee the safe operation of 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 the system's autonomy.

Primarily, Xu et al. (2017), have made a classification of fault diagnosis systems by dividing them into:

- *Knowledge-driven models. S*ystem with a small number of inputs and outputs, easily to model, but only for specific types of failures.
- *Data-driven models.* These methods can increase the diagnostic accuracy and the degree of automation using data mining in historical data.
- *Value-driven methods.* Similar to data-driven, but this type of fault diagnosis system uses big data and big data analytical methods to detect significant values that are not easily detected by traditional methods.

In the next section, the design and implementation of the autonomic cycle for self-supervising is detailed.

5.2 Design of the Autonomic cycle for 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). Furthermore, this autonomic cycle consists of three data analytics tasks, which are detailed below:

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

The characteristics of this task are listed in Table 5.1. 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 in order to detect failures.

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

Table 5.1. Data Analytics Task 1 Characteristics

5.2.2 Task 2: Build a predictive model of product quality 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 products. This model detects failures before each product enters in the manufacturing line, and allows repairing the system in order to fix the system. The characteristics of this task are shown in Table 5.2.

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 5.2. Data Analytics Task 2 Characteristics

5.2.3 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 related to 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 5.3.

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

Table 5.3. Data Analytics Task 3 Characteristics

The components' diagram of the autonomic cycle for self-supervising prototype application is shown in Figure 5.1. The Business Process is the component that is supervised (the managed resource). 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.

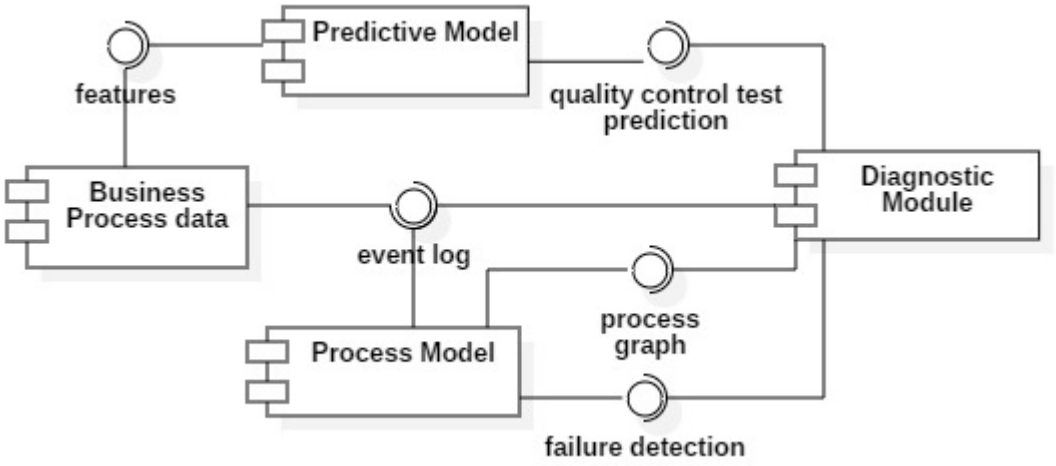

Figure 5.1. Autonomic cycle of self-configuring (component diagram).

5.3 Case study

5.3.1 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 station's goals, 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 this dataset 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, and it adds another difficulty to the knowledge extraction process. 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.

5.3.2 Implementation of the autonomic cycle for self-supervising

In this sub-section, it is detailed the implementation of the autonomic cycle described in Section 5.2. The macro algorithm used to implement this autonomic cycle, as well as the technological tool used, are shown in Table 5.4.

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++.

Table 5.4. Macro-Algorithms to Implement the Self-Supervising Autonomic Cycle

Task 1. For this task, it 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 Figure 5.2 (see Appendix A for details on how to transform this data).

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

Figure 5.2. 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 processing the

auto-part. In the Bosch dataset, the format of that value is a decimal number, and it does not correspond with the required format, as shown in Figure 5.2.

- 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.

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 unique file with all the process model information (see Appendix A for details). The information saved to this model is shown in Figure 5.3. 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 calculate that value).

```
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 5.3. Bosch production line process model.

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. Another Python script builds this predictive model (see Appendix A). This model could 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 employed for this kind of machine learning models (see Appendix B6.3Appendix B for details about the confusion matrix for each classifier), such as precision, recall, f1-score, accuracy, Matthew's correlation coefficient (MCC), and the Receiver operating characteristic (ROC) curves. Table 5.5 shows a summary of those metrics.

Table 5.5. 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-RF 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 to predict 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, Figure 5.4 presents the ROC curves for both classifiers.

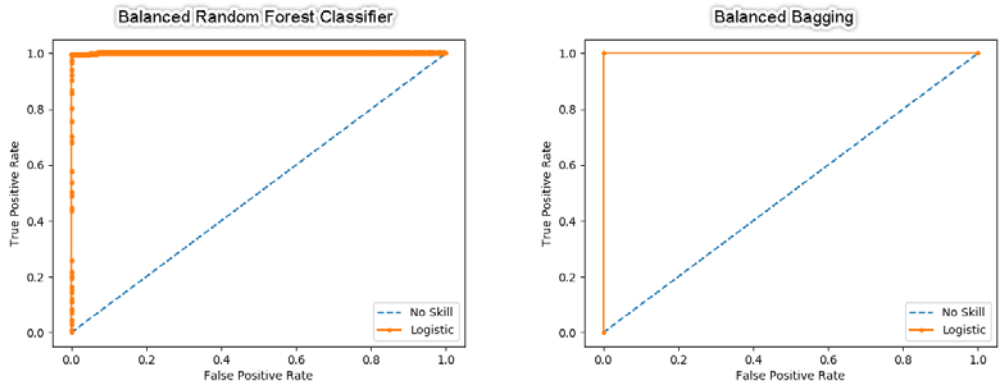

Figure 5.4. Comparison of B-RF and B-B classifiers using the ROC curves.

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 supervisory system better 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). Figure 5.5 describes how the

people actor can interact with this prototype.

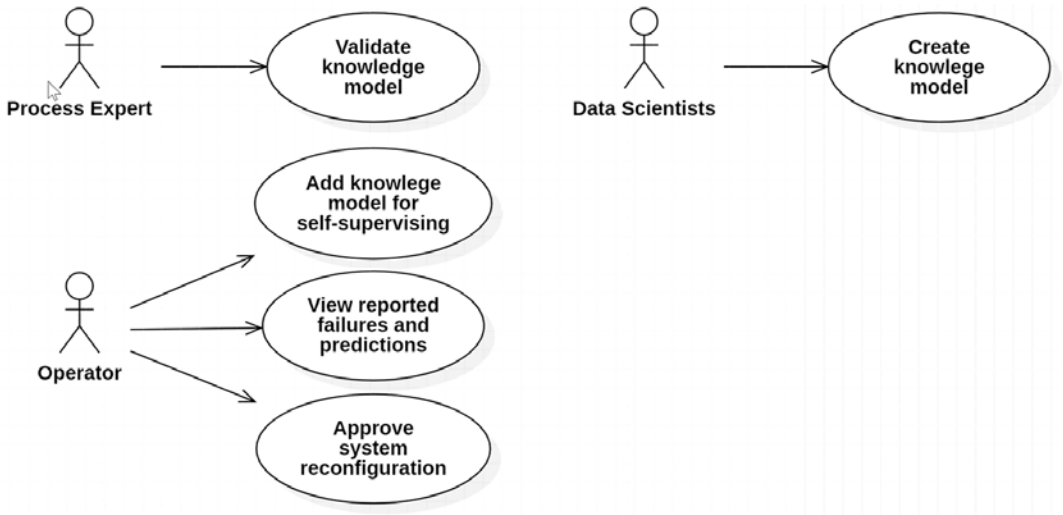

Figure 5.5. Use case of the people actor.

The Data Scientist is in charge of creating the everything-mining models by applying different mining techniques. For this particular research, he generated 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 autonomous 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.

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. Figure 5.6 shows, more specifically, the functionality of the autonomic cycle for 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 continually sends data to the Message queue (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 treat 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), and the autonomic cycle of self-healing is invoked (message 16) in order to repair the manufacturing process.

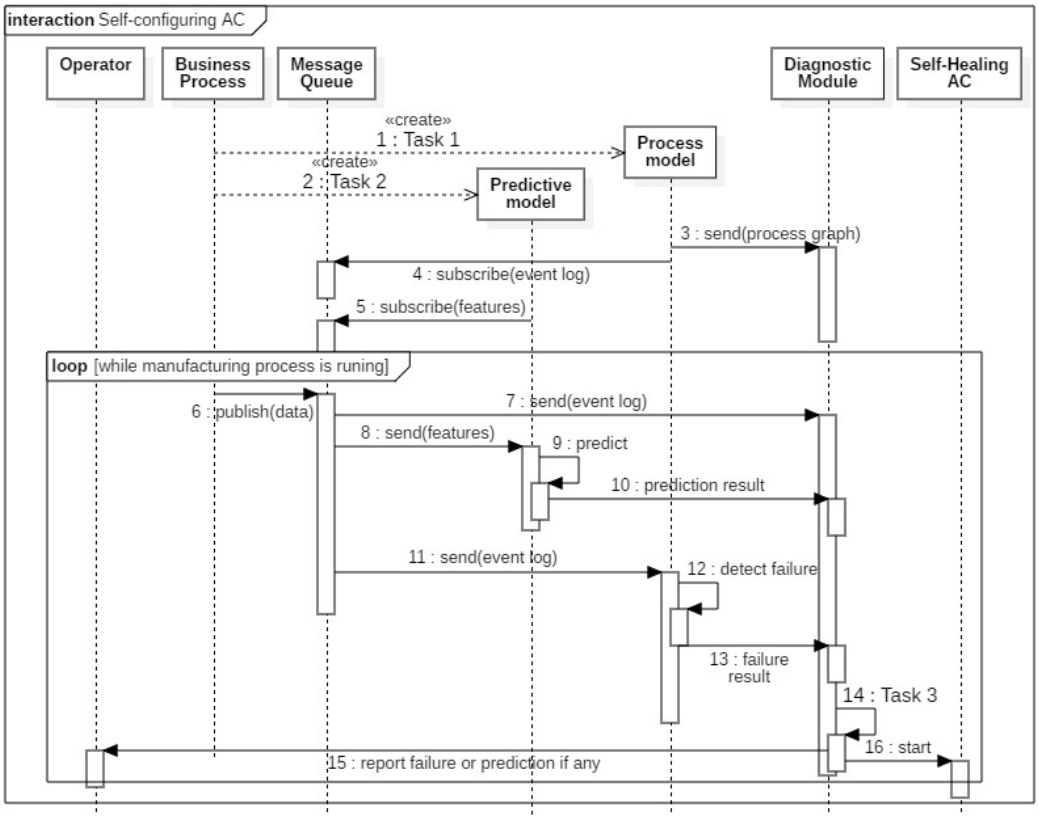

Figure 5.6. Autonomic cycle of self-supervising (sequence diagram).

As can be seen, the operator is not involved in the decision-making procedure because the process is autonomic, and also, because this step corresponds only to the failure detection phase. Moreover, 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. Figure 5.7 shows a screenshot of the prototype application of the self-supervisory system. This prototype application allows the operator to define which 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 application allows the operator to start, stop, pause, restart, and accelerate the test of the manufacturing process. Besides, this prototype application diagnoses the system (predicts quality control test results and detect performance issues) and invokes the autonomic cycle for self-repairing autonomously when required. In this way, it implements our autonomic cycle of self-supervising.

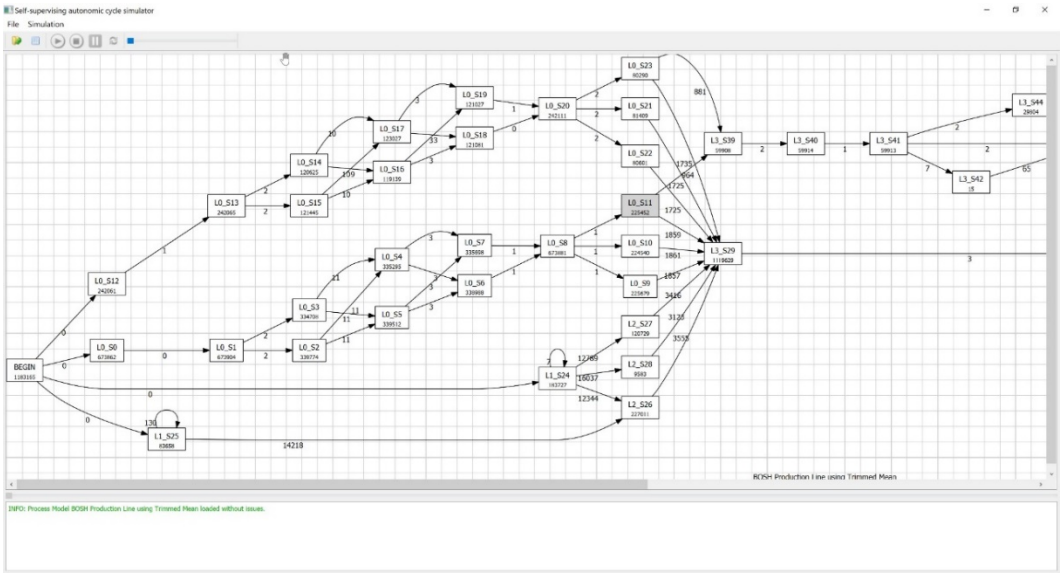

Figure 5.7. Self-supervising autonomic cycle prototype application.

5.4 Result

In this subsection, it will be discussed the result obtained in each phase of the supervisory system implementation.

5.4.1 Task 1

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

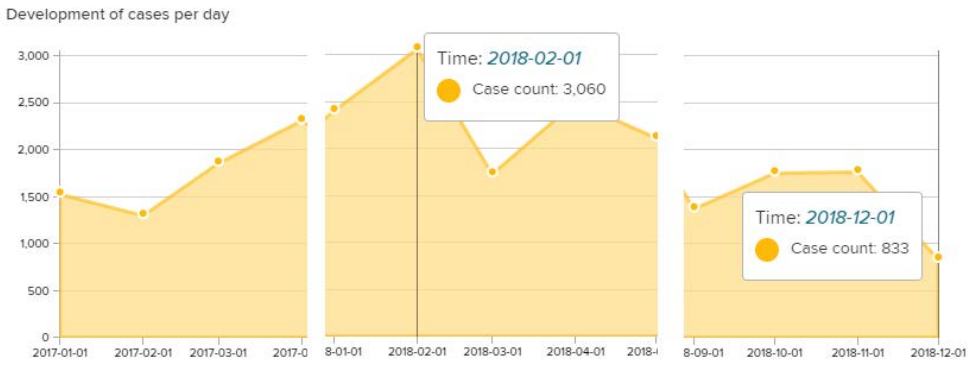

Figure 5.8. Min and max number of cases produced daily.

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 Figure 5.8). Another essential metric regarding the global performance of the manufacturing process is the average throughput time (See Figure 5.9), it indicates that each product is produced in about 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.

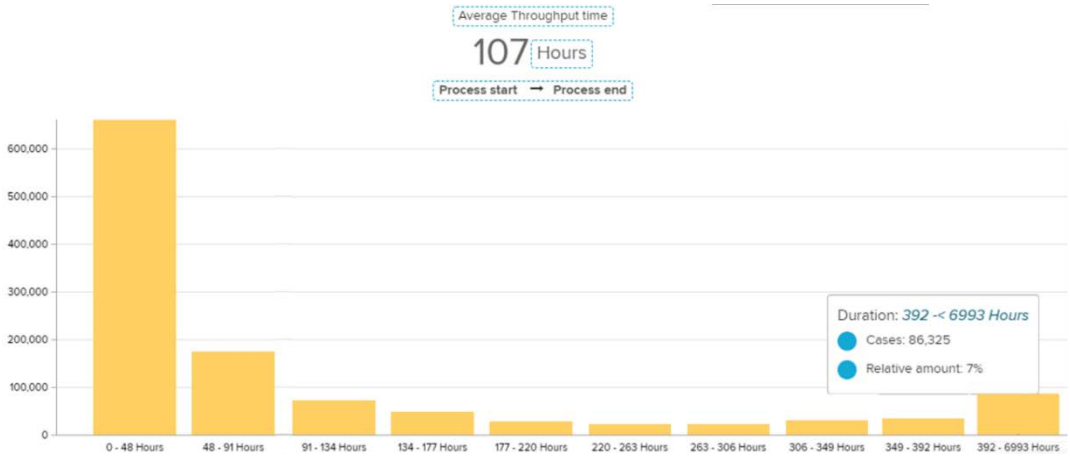

Figure 5.9. Throughput time.

In Figure 5.10, the stations that present bottleneck are shown. E.g., station 24 takes 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 most 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 to know what stations need special attention.

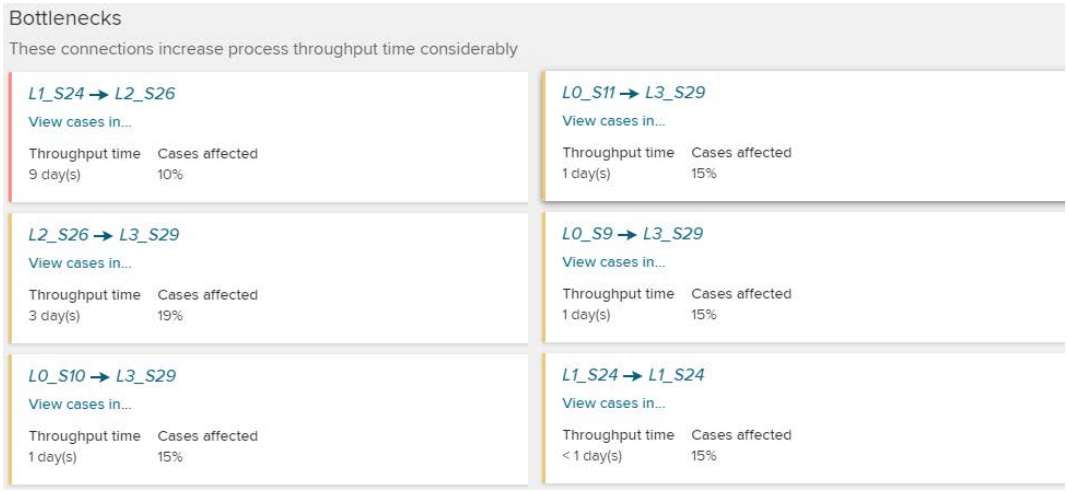

Figure 5.10. Stations with bottlenecks.

Likewise, Figure 5.11 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. The previous statement confirms the insights that Mangal and Kumar (2016) discover about station 32 using data mining: "station 32 has the highest error rate. Also, station 32 does not process many products; hence its impact on the production yield is minimal".

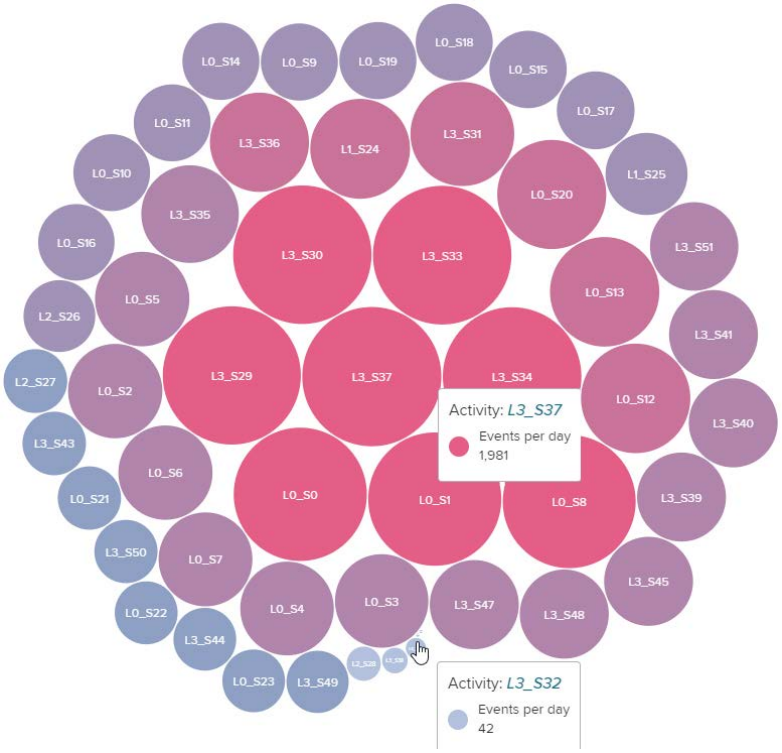

Figure 5.11. Stations' workload.

Regarding the Bosch manufacturing process, Figure 5.12 shows the layout of this production process. Principally, each rounded rectangle represents one actor (a Bosch station), and lines represent transitions or paths (edges). Also, 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. Furthermore, Figure 5.12 details that although there are four production lines, they are not independent of each other.

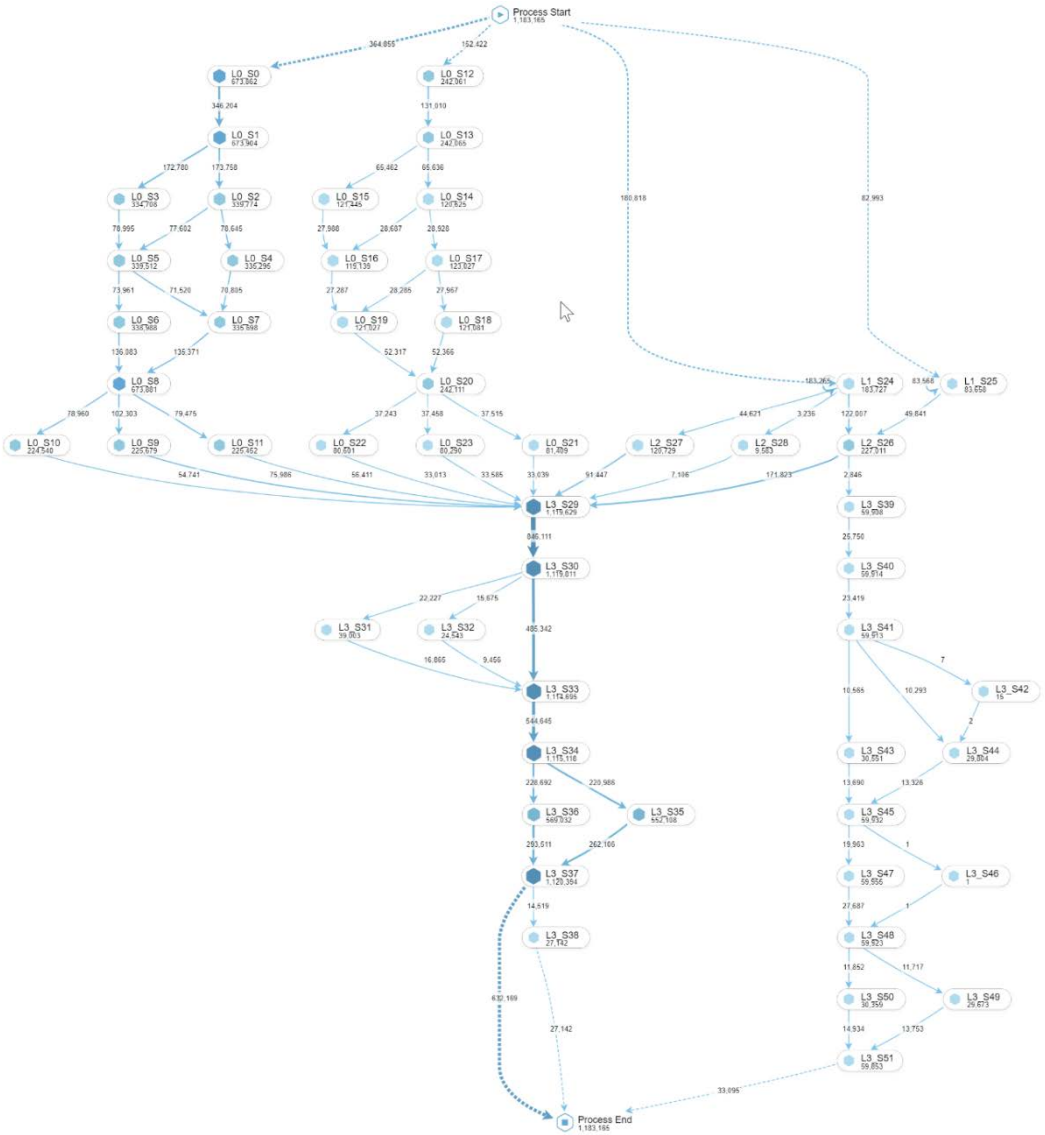

Figure 5.12. Bosch manufacturing process' layout.

5.4.2 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 prediction and failure detection in a dashboard so that the operator can check what is happening in the manufacturing process (see Figure 5.13). The red circle represents the auto-parts being manufactured, and its movement thought the production line.

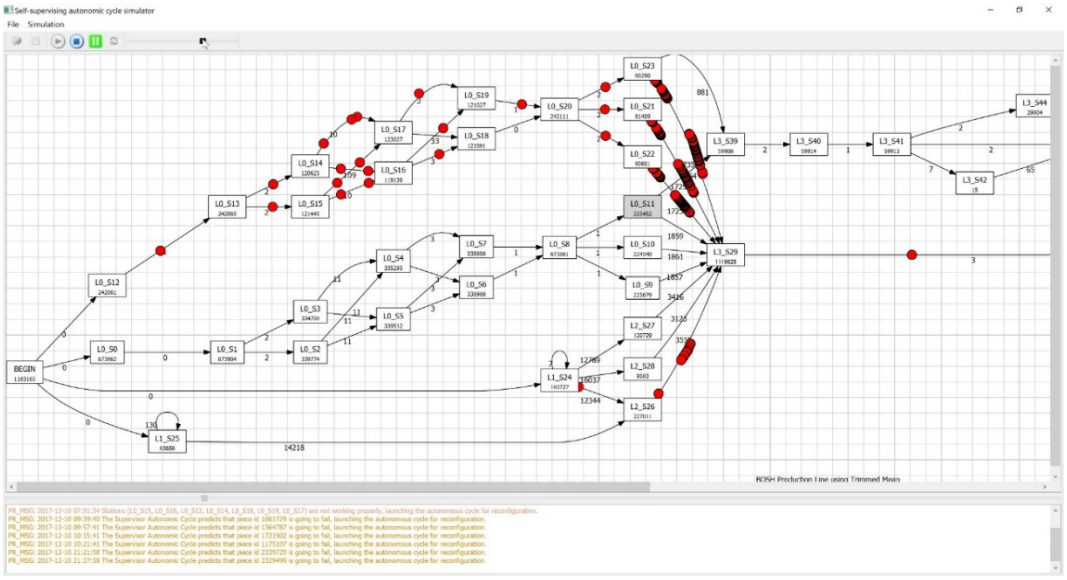

Figure 5.13. Bosch manufacturing process.

5.4.3 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. Figure 5.14 shows the control messages about the failures and quality control test predictions.

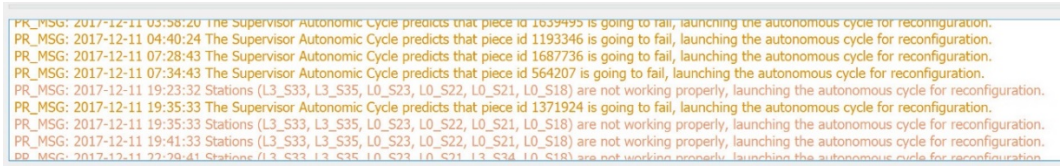

Figure 5.14. Self-supervision dashboard.

For instance, Figure 5.14 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). Figure 5.15 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.

Figure 5.15. Average time for failure and predictions.

5.5 Comparison with other researches

On this section, we will use the next set of criteria to compare our approach with previous researches (see Table 5.6):

1. 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 different failures.

2. 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.

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 collected.

5. Scalability: Indicates if the supervisory system supports the future inclusion of new mining-techniques or self-* properties.

6. The autonomy of the supervisory system: This element specifies the level of autonomy (None, Low, Medium, High, Fully autonomous) of the supervisory

system.

As can be seen from Table 5.6, the supervisory systems that were built, 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, any 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.

	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

Table 5.6. Related works' characteristics

Concerning the Number of Everything-mining techniques used by the supervisory system, our self-supervising autonomic cycle uses two Everything-mining techniques. Explicitly, it has used process mining and data mining in order to create a process model and a predictive model that help in getting a diagnostic of the current status of the manufacturing process and launching the corresponding autonomic cycle for 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. However, 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's 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. Mainly, 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. This information was useful for failure detection.
- Detecting the relationship between actors and production lines. Besides, it detects that the production lines are not independent and that Line 0 consists of two sub-lines.

This information is not easy to get using another data analytics technique. 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 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. In Task 1 we got historical information about the stations' performance. 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 for 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 other self-* properties incrementally to gain more and more autonomy with the inclusion of new autonomic cycles of data analytical tasks.
- Finally, the use of the Everything-mining techniques ensures the scalability of the system in order to treat any source of information.

Finally, regarding the Autonomy of the supervisory system, as can be seen from Figures 5.6, 5.13 and 5.14 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 could not 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 do not have any metric that could be used for comparison.

5.6 Summary

This chapter has shown the implementation of the autonomic cycle of self-supervising as a 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.

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

- Capacity to support several Everything-mining techniques. Each mining technique improves the knowledge of the system, and by consequence, the decision-making processes.
- Integration of all actors involved in manufacturing processes, such as Thing, Data, People, and Services.
- Support for self-* properties that are added gradually, guaranteeing in this sense the scalability of the system.
- Support for real-time analysis and decision-making.
- Capacity of diagnosing the system and start repairing it autonomously.

Chapter 6

Conclusions and Perspectives

Contents

6.1 Introduction

In this chapter, we summarized the results of our research project by presenting a set of contributions in Section 6.2. In this sense, we offer the conclusion of our work, along with the lessons learned in this Ph.D. thesis. Finally, in Section 6.3, we present the perspectives and highlight new future research directions.

6.2 Conclusions

Industry 4.0 is a concept still in development that will transform not only the way like goods and services are produced but our daily life. It is expected that Industry 4.0 reaches its maximum potential in the next coming years. In the meanwhile, many kinds of research must be pushed to solve the challenges around this new concept. Principally, in this thesis project, we were dealing with the integration challenges and interoperability issues, in order to allow the actors of manufacturing processes to autonomously connect, communicate, coordinate, cooperate, and collaborate. Our solution combines three main elements: The IoE, the everything-mining, and the autonomic computing in order to incrementally provide contributions to solve the integration challenges of Industry 4.0.

Problem description: Autonomous processes in Industry 4.0

One fundamental feature of production processes in the context of Industry 4.0 is autonomy. The main goal, in this case, is to leave the actors involved in the manufacturing process to act (plan, control, make-decision) by themselves. In consequence, many efforts are required in order to achieve this feature. In this thesis, we have proposed an incremental approach to solve the Industry 4.0 challenges and enable the integration and interoperability of actors in manufacturing processes. Our

solution not only allows the actors to autonomously connect and communicate by using technologies like IoE and MAS, but to create more elaborated processes like coordination, cooperation, and collaboration, which are incrementally added as self-* properties, using the autonomic computing paradigm and everything-mining techniques. In this sense, each time that a new self-* property is added, the system gains more autonomy.

Furthermore, the everything-mining techniques used in this thesis were fundamentals, in order to use information from different types of sources and to break with the heterogeneity of actors. Additionally, the everything-mining techniques allowed us to create the knowledge bases needed for decision making in all levels of the 5C stack.

Connection and communication levels

As discussed in Chapter 3, connection and communication are fundamental processes in any cloud-based system, such as Intelligent environment, Smart rooms, Smart cities, and Industry 4.0. Additionally, connection and communication are a starting point to allow coordination, cooperation, and collaboration processes in any system. In contrast, MAS are vastly used to build Cyber-physical systems, in Industry 4.0 context, because it incorporates well-defined protocols in all levels of the 5C stack. However, MAS and Cloud systems represent another challenge of integration. In that sense, this thesis proposed a solution of integration MAS, Cloud, and Fog computing, which was tested in domains like Smart rooms, Smart cities, and Industry 4.0. Particularly, this solution lets agents and cloud-computing services to naturally and autonomously interoperate so that agents and services do not need to worry about message translations or any other communication details.

Coordination, cooperation, and collaboration levels

Autonomous coordination, cooperation, and collaboration processes are the final components in the 5C stack levels. They will allow the actors of the manufacturing process to interoperate in order to reach collective or individual goals oriented to build goods or provide services. At these levels, actors will be able to organize themselves. It means to create a production plan, to start a conversation with another actor, to start a task execution, in short words, to make decisions according to a plan for coordination, cooperation, or collaboration. Notably, in this thesis, we proposed a framework for the integration and interoperability of actors in Industry 4.0 that promotes processes for coordination cooperation and collaboration throughout the incorporation of self-*properties. Moreover, we proposed three autonomic cycles of data analytics that actors can autonomously use for the coordination of the processes in manufacturing contexts. These three autonomic cycles promote properties of self-planning, self-managing, self-supervising, and self-healing to manufacturing processes, and use multiple everything-mining techniques to create the knowledge bases needed by actors to make decisions.

Implementation approach

We culminated our research project with the implementation of the proposed framework for the integration and interoperability of actors in Industry 4.0. Particularly, our proposed framework was tested using a case study corresponding to the Bosh manufacturing process. In this case, we developed one of the three autonomic cycles of data analytical tasks proposed. This autonomic cycle brings self-supervising functionality to the manufacturing process.

The MIDANO methodology drove the design of the autonomic cycle for self-supervising. Moreover, in this phase, we used two everything-mining techniques. Firstly, we used a Process mining technique that helps us to get enough insight about the manufacturing process, like the process layout, the processing time of each station, the global throughput time, bottlenecks, and others. With this information, a model of the production process was built and used to make decisions oriented to detect failures during the execution of the production process and to launch the automatic cycle for self-healing. Next, we built a predictive model based on the quality control test parameter included in the manufacturing data. This model was built using machine learning techniques and allows us to predict the future failures of the quality control test, so that the system can self-healing by reconfiguring the manufacturing process (one of the others autonomic cycle).

The knowledge models built using the everything-mining techniques allowed us to develop a value-driven supervisory system, which can replay the Bosh manufacturing process to verify the validity of our proposal. Moreover, the comparison with previous works shows that our solution was satisfactory and represents a significant contribution in this context.

6.3 Perspectives

The contributions achieved during this work introduced several novelties regarding the Industry 4.0 context. Hence, this study opens the doors to a diversity of new research directions that can complement and polish our approach:

1. Support for digital twins.

   The result of this thesis can be extended by implementing the proposed framework using a digital twin. In that sense, the Arcadia methodology (Roques, 2016) can be used to design a Capella-based autonomic manufacturing system for Industry 4.0. Next, a simulated environment based on this design can be created using the ROS Industrial (2018) middleware.

2. Implement the self-planning, self-manage, and self-healing properties of the coordination process.

   Another perspective derived from this thesis is related to implement the specific autonomic cycles designed in Chapter 4, to allow manufacturing processes to be fully self-coordinated.

3. Design and implement autonomous processes for self-cooperation and self-collaboration.

   The proposed framework discussed in Chapter 4, support not only coordination processes, but also self-cooperation and self-collaboration processes. However, it still needed to design and implement the autonomic managers that will cope with the self-collaboration and self-cooperation properties, in order to turn manufacturing processes into a fully integrated and autonomous manufacturing system.

4. Improve the self-supervising application.

   Adding new features to the current prototype application developed in this research is another interesting future research. The idea is to add support for other models created using the everything-mining techniques. Thus, the operator can decide which models to incorporate in order to detect and predict failures. Also, a process mining algorithm can be added for this application to avoid using external actors.

5. Incorporate new everything-mining techniques to the framework

   The present thesis used two everything-mining techniques: process mining and data mining. In the future, it could be interesting to add a thing mining as DEKG (Y. Xu et al., 2017), to get a proper failure diagnostic on each station, and increase in this way, the number of failures that this system can detect. Finally, it must be a challenge to add other mining techniques like service mining, sentiment analysis, and others, and use the collected information to improve the autonomy of manufacturing processes.

6. Incorporate new communication protocols to the MAS-SOA-Fog middleware.

   In this sense, another perspective must be oriented to add other common cloud communication languages, so that the "Message Translator" component can make the corresponding transformations, in order to allow agents to communicate with cloud services that share data in formats like JSON, XML, etc.

Appendix A

Macro-Algorithm

This appendix presents the macro algorithms used to build the Everything-mining models for self-supervision.

A.1 Create an event log from process data

This macro algorithm transforms the Bosh input training data into an event log file. Firstly it takes the training data set in a specific format and iterate over each row, getting the corresponding columns like the starting time, station name, and event id. Next, this information is put into the output event log file in the correct format. This macro algorithm is detailed in Table A.1

Macro algorithm

Input data: {trainingData} → eventLog
Procedure:
 	
currentDate = Date.currentDate
open file (trainingData) for input
open file (outputFile) for output
outputFile.writeLine(['Id', 'Activity', 'Date']);
index = 0
stopIndex = -1
 	
for each row in trainingData:
  if stopIndex > 0 and index == stopIndex:
  break
  for column, value in row.items():
  if(column == 'Id' or not value):
  continue
  seconds = int(float(value) * 10 * 60 * 60);
  date = currentDate + seconds
  col = column.split("_")
  event = "_".join(col[0:2])
  f = col[2][1:]
  outputFile.writeLine([row['Id'], event, date.format('%Y-%m-%d %H:%M:%S')])
  index += 1
outputFile.drop_duplicates()
outputFile.close()
trainingData.close()

Table A.1. Macro algorithm for event log discover from the process data.

A.2 Creating the manufacturing process model

This macro algorithm takes the production process data and generate the process model that can be used for decision-making.

Macro algorithm

Input data: {processDataFileTimes, processDataFileCases, processDataFileActors, throughputTime, modelName } → eventLog

Procedure:

```
times_df = None
cases_df = None
actors_df = None
visibility_df = None
times_df = open(processDataFileTimes) for input as CSV
cases_df = open(processDataFileCases) for input as CSV
actors_df = open(processDataFileActors) for input as CSV
num_c_times = times_df.colums
num_c_cases = cases_df.colums
actorsStr = actors_df.headerRow
actorsList = actorsStr.split(",")
beginActors = [s for s in actorsList if s.startswith("+")]
endActors = [s for s in actorsList if s.startswith("-")]
totalCases = 0
df_combined = pd.DataFrame(columns=['transitions:'])

for ( idxRow, s1 ), ( _, s2 ) in zip( times_df.iterrows(), cases_df.iterrows() ) :
    concat = '';

    for ( idxCol, v1 ), ( _, v2 ) in zip( s1.iteritems(), s2.iteritems() ) :
      if idxCol == 'actor':
        continue
      if (not pd.isna(v1)) and (not pd.isna(v2)):
        if "+" + str(s2.head(1).item()) in beginActors:
          totalCases = totalCases + v2
        visibility = '0'
        if visFileExists:
          vis = visibility_df[idxCol].iloc[idxRow]
          visibility = '0' if pd.isna(vis) or vis != 1  else '1'
        concat = concat + ";".join([idxCol,str(v1),str(v2), str(visibility)])+"!"
    df_combined.loc[idxRow] = s1[0] + '=' + concat[:-1]

modelFile = open(baseFileName + "_model.pg", 'w') for output
modelFile.write("process:"+modelName+'\n')
modelFile.write("actors:"+actorsStr+'\n')
modelFile.write("process_throughput_time:"+str(throughputTm)+'\n')
modelFile.write("total_cases:"+str(int(totalCases))+'\n')
modelFile.write("transitions:"+'\n')
for v1 in df_combined['transitions:']:
  modelFile.write(v1+'\n')
modelFile.close()
```

Table A.2. Macro algorithm to create the process model from the process mining output files

This macro algorithm presented in Table A.2 combines the inputs files of station times, processed products by station, actors, and the total throughput time into a single file that can be read by the autonomic cycle and use the information for decision making.

A.3 Process model training macro algorithm

Macro algorithm

Input data: {trainingFile} → CNN Model
Procedure:

```
def buildClassifier( ninputs ):
    #create model
    model = Sequential()
    #add layers to model
    model.add(Dense(ninputs, kernel_regularizer=l2(0.001), activation='relu',
input_shape=(ninputs,)))
    model.add(Dense(ninputs*2+1, activation='relu'))
    model.add(Dense(1))
    model.compile(optimizer='adam', loss='mean_squared_error', metrics=['accuracy'])

    return model

def trainModel( trainfile, output_dir='./', num_epochs=30 ):
    train_df = pd.read_csv(trainfile)
    train_df = train_df.replace(np.nan, 0)
    train_input = train_df.drop(columns=['Id','Response'])
    train_output = train_df[['Response']]
    ninputs = train_input.shape[1]
    y = train_output.Response.tolist()
    class_weights = class_weight.compute_class_weight('balanced', np.unique(y), y)
    class_weight_dict = dict(enumerate(class_weights))

    # Saves the model weights after each epoch if the validation loss decreased
    now = datetime.now()
    nowstr = now.strftime('bosch-%Y%m%d%H%M%S')
    now = os.path.join( output_dir, nowstr)
    os.makedirs( now, exist_ok=True )
    # Create our callbacks
    savepath = os.path.join( now, 'e-{epoch:03d}-vl-{val_loss:.3f}-va-{val_acc:.3f}.h5' )
    checkpointer = ModelCheckpoint(filepath=savepath, monitor='val_acc', mode='max',
verbose=1, save_best_only=True)
    # patient early stopping
    es = EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=3)
    classifier = buildClassifier(ninputs)
    classifier.fit(train_input, train_output, validation_split=0.2, class_weight=class_weight_dict,
                   epochs=num_epochs,  callbacks=[es, checkpointer])

  return classifier
```

Table A.3. Predictive model training macro algorithm

The macro algorithm presented in Table A.3 is in charge of building the predictive model using the process model's training data. This algorithm creates a convolutional neural network with three layers.

Appendix B

Confusion matrix

This appendix shows the confusion matrix for each classifier after predicting the testing data, which contains 100.000 testing cases.

B.1 Balanced Random Forest Classifier (B-RF).

6.3Table B.1 shows that there are no false-positive cases; it means that any product was classified to pass the quality control test when it fails in really. However, seven parts that will pass the quality control test were classified incorrectly to fail.

Predicted		True/Actual	
		Pass	**Fail**
	Pass	99401	0
	Fail	7	592

Table B.1. B-RF Confusion matrix

B.2 Balanced Bagging Classifier (B-B).

Table B.2 shows that there are no false-positive cases; it means that any product was classified to pass the quality control test when it fails in really. However, five parts that will pass the quality control test were classified incorrectly to fail.

Predicted		True/Actual	
		Pass	**Fail**
	Pass	99403	0
	Fail	5	592

Table B.2. B-B Confusion matrix

B.3 Convolutional Neural Network (CNN).

Table B.3 shows that 589 parts that will fail the quality control test were classified incorrectly to pass it. Similarly, 612 parts that will pass the quality control test were classified to fail it. In this case, it is easy to notice that this classifier is not getting good results in classifying both classes, and this will represent a significant problem concerning the time used to reconfigure the manufacturing process.

Predicted		True/Actual	
		Pass	**Fail**
	Pass	98796	589
	Fail	612	3

Table B.3. CNN Confusion matrix

B.4 Linear Discriminant Analysis (LDA).

Table B.3 shows that 688 parts that will fail the quality control test were classified incorrectly to pass it. Similarly, 456 parts that will pass the quality control test were classified to fail it. The results of this classifier are quite similar to the results of CNN and will represent a significant issue regarding the time used to reconfigure the manufacturing process.

Predicted		True/Actual	
		Pass	**Fail**
	Pass	98728	688
	Fail	456	128

Table B.4. LDA Confusion matrix

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., 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., Garcès, A., Gallego, N., Gutiérrez, J., Gomez, J., & García, Á. (2019). Autonomic Management Architecture for Multi-HVAC Systems in Smart Buildings. *IEEE Access*, *7*, 123402–123415. https://doi.org/10.1109/ACCESS.2019.2937639

Aguilar, J., Jerez, M., Mendonca, M., & Sanchez, M. (2016). MiSCi: Autonomic Reflective Middleware for Smart Cities. In R. Valencia-García, K. Lagos-Ortiz, G. Alcaraz-Mármol, J. del Cioppo, & N. Vera-Lucio (Eds.), *Procedings of Technologies and Innovation: Second International Conference, CITI 2016, Guayaquil, Ecuador* (pp. 241–253). Springer International Publishing. https://doi.org/10.1007/978-3-319-48024-4_19

Aguilar, J., Mendoça, M., Jerez, M., & Sanchez, M. (2017c). Emergencia ontológica basada en análisis de contexto, como servicio para ambientes inteligentes. *DYNA Rev.Fac.Nac.Minas*, *84*(200), 28–37. https://doi.org/10.15446/dyna.v84n200.59062

Aguilar, J., Sanchez, M., Cordero, J., Valdiviezo-Díaz, P., Barba-Guamán, L., & Chamba-Eras, L. (2017d). 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. (2017e). 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

Aguilar, J., Sanchez, M., Vadiviezo, P., & Cordero, J. (2015a). Mecanismos de Coordinación en un Salón Inteligente. *6TO CONGRESO IBEROAMERICANO DE ESTUDIANTES DE INGENIERÍA ELÉCTRICA (VI CIBELEC 2015)*, 31–40. http://repositorio.educacionsuperior.gob.ec/handle/28000/3983

Aguilar, J., Valdiviezo, P., Cordero, J., & Sanchez, M. (2015b). Conceptual design of a smart classroom based on multiagent systems. *Proceedings on the International Conference on Artificial Intelligence (ICAI)*, 471–477. http://repositorio.educacionsuperior.gob.ec/handle/28000/3987

Al-Ayyoub, M., Jararweh, Y., Daraghmeh, M., & Althebyan, Q. (2015). Multi-agent based dynamic resource provisioning and monitoring for cloud computing systems infrastructure. *Cluster Computing*, *18*(2), 919–932. https://doi.org/10.1007/s10586-015-0449-5

Amadeo, M., Molinaro, A., Paratore, S. Y., Altomare, A., Giordano, A., & Mastroianni, C. (2017). *A Cloud of Things framework for smart home services based on Information Centric Networking*. 245–250. https://doi.org/10.1109/ICNSC.2017.8000099

Apache Software-Foundation. (2008). *Apache CXF*. Apache CXF™: An Open-Source Services Framework. https://cxf.apache.org/

Archimède, B., Memon, M. A., & Ishak, K. (2017). Combining multi-agent model, SOA and ontologies in a distributed and interoperable architecture to manage multi-site production projects. *International Journal of Computer Integrated Manufacturing*, *30*(8), 856–870. https://doi.org/10.1080/0951192X.2016.1224389

Auger, A., Exposito, E., & Lochin, E. (2017). Survey on Quality of Observation within Sensor Web systems. *IET Wireless Sensor Systems*, *7*(6), 163–177. https://doi.org/10.1049/iet-wss.2017.0008

Bahrin, M. A. K., Othman, M. F., Azli, N. H., & Talib, M. F. (2016). Industry 4.0: A review on industrial automation and robotic. *Jurnal Teknologi*, *78*(6–13), 137--143.

Bohuslava, J., Martin, J., & Igor, H. (2017). TCP/IP protocol utilisation in process of dynamic control of robotic cell according industry 4.0 concept. *2017 IEEE 15th International Symposium on Applied Machine Intelligence and Informatics (SAMI)*, 000217–000222. https://doi.org/10.1109/SAMI.2017.7880306

Bonomi, F., Milito, R., Zhu, J., & Addepalli, S. (2012). Fog computing and its role in the internet of things. *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, 13–16.

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

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

Cavalieri, S., Salafia, M. G., & Scroppo, M. S. (2019). Towards interoperability between OPC UA and OCF. *Journal of Industrial Information Integration*, *15*, 122–137. https://doi.org/10.1016/j.jii.2019.01.002

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

Chang, V., & Wills, G. (2016). A model to compare cloud and non-cloud storage of Big Data. *Future Generation Computer Systems*, *57*, 56–76. https://doi.org/10.1016/j.future.2015.10.003

Chen, K., Hsu, S., Jhang, J., & Lin, C. (2017). *INTERNET OF THINGS SECURITY APPLIANCE* (Patent No. United States Patent Application 20170289176 Kind Code: A1). http://www.freepatentsonline.com/y2017/0289176.html

Chen, X., Yang, P., Qiu, T., Yin, H., & Ji, J. (2017). IoE-MPP: A mobile portal platform for internet of everything. *Journal of Intelligent & Fuzzy Systems*, *32*(4), 3069–3080. https://doi.org/10.3233/JIFS-169250

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

Commission Académique. (2017). *Thèse par articles*. Ecoles Doctorales Sciences Exactes et Applications de la Université de PAU et les Pays de l'Adour. https://ed-sea.univ-pau.fr/_attachments/nouvel-article-2/Th%25C3%25A8se%2520par%2520article%2520Commission%2520Acad%25C3%25A9mique%252028sept2017.pdf?download=true

Decker, G., Kopp, O., Leymann, F., & Weske, M. (2007). BPEL4Chor: Extending BPEL for Modeling Choreographies. *IEEE International Conference on Web Services (ICWS 2007)*, 296–303. https://doi.org/10.1109/ICWS.2007.59

Deloitte Consulting. (2017). *The smart factory*. Delloite University Press. https://www.google.co.ve/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&ved=0ahUKEwjN3eWstv_XAhWKOxQKHfk8AaIQFghHMAI&url=https%3A%2F%2Fdupress.deloitte.com%2Fcontent%2Fdam%2Fdup-us-en%2Farticles%2F4051_The-smart-factory%2FDUP_The-smart-factory.pdf&usg=AOvVaw3XZS9-BuP0iQ9EL7EHPRgw

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

Drăgan, I., Selea, T., & Fortiş, T.-F. (2017). Towards the Integration of a HPC Build System in the Cloud Ecosystem. *Complex, Intelligent, and Software Intensive Systems*, 916–925. https://doi.org/10.1007/978-3-319-61566-0_87

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

Exposito, E. (2013). *Advanced Transport Protocols: Designing the Next Generation*. John Wiley & Sons.

Exposito, E., & Diop, C. (2014). *Smart SOA Platforms in Cloud Computing Architectures*. Wiley-ISTE. https://doi.org/10.1002/9781118761489

Faheem, M., & Gungor, V. C. (2017). Energy efficient and QoS-aware routing protocol for wireless sensor network-based smart grid applications in the context of industry 4.0. *Applied Soft Computing*. https://doi.org/10.1016/j.asoc.2017.07.045

Ferrera, E., Rossini, R., Baptista, A. J., Evans, S., Hovest, G. G., Holgado, M., Lezak, E., Lourenço, E. J., Masluszczak, Z., Schneider, A., Silva, E. J., Werner-Kytölä, O., & Estrela, M. A. (2017). Toward Industry 4.0: Efficient and Sustainable Manufacturing Leveraging MAESTRI Total Efficiency Framework. *Sustainable Design and Manufacturing 2017*, 624–633. https://doi.org/10.1007/978-3-319-57078-5_59

Fu, X., Bultan, T., & Su, J. (2004). Analysis of Interacting BPEL Web Services. *Proceedings of the 13th International Conference on World Wide Web*, 621–630. https://doi.org/10.1145/988672.988756

Fuksa, M. (2014). *Methods and Tools for Intelligent ESB* [Master]. Czech Technical University in Prague.

García Coria, J. A., Castellanos-Garzón, J. A., & Corchado, J. M. (2014). Intelligent business processes composition based on multi-agent systems. *Expert Systems with Applications*, *41*(4, Part 1), 1189–1205. https://doi.org/10.1016/j.eswa.2013.08.003

Giordano, A., Spezzano, G., & Vinci, A. (2016). Smart Agents and Fog Computing for Smart City Applications. *Smart Cities*, 137–146. https://doi.org/10.1007/978-3-319-39595-1_14

Gökalp, E., Şener, U., & Eren, P. E. (2017). Development of an Assessment Model for Industry 4.0: Industry 4.0-MM. *Software Process Improvement and Capability Determination*, 128–142. https://doi.org/10.1007/978-3-319-67383-7_10

Goossens, J., & Richard, P. (2017). *Handbook of Cyber-Physical Systems*. Multiprocessor Real-Time Scheduling. http://hdl.handle.net/2013/ULB-DIPOT:oai:dipot.ulb.ac.be:2013/250127

Greenwood, D., Buhler, P., & Reitbauer, A. (2005). Web service discovery and composition using the web service integration gateway. *The 2005 IEEE International Conference on E-Technology, e-Commerce and e-Service.*, 789–790.

Greenwood, D., & Calisti, M. (2004). Engineering Web service—Agent integration. *2004 IEEE International Conference on Systems, Man and Cybernetics*, *2*, 1918–1925 vol.2. https://doi.org/10.1109/ICSMC.2004.1399962

Gupta, H., Dastjerdi, A. V., Ghosh, S. K., & Buyya, R. (2016). iFogSim: A Toolkit for Modeling and Simulation of Resource Management Techniques in Internet of Things, Edge and Fog Computing Environments. *Software: Practice and Experience*, *47*(9), 1275–1296. https://doi.org/10.1002/spe.2509

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

Heiss, M., Oertl, A., Sturm, M., Palensky, P., Vielguth, S., & Nadler, F. (2015). Platforms for industrial cyber-physical systems integration: Contradicting requirements as drivers for innovation. *2015 Workshop on Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES)*, 1–8. https://doi.org/10.1109/MSCPES.2015.7115405

Heldal, R., Pelliccione, P., Eliasson, U., Lantz, J., Derehag, J., & Whittle, J. (2016). Descriptive vs Prescriptive Models in Industry. *Proceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems*, 216–226. https://doi.org/10.1145/2976767.2976808

Hofmann, E., & Rüsch, M. (2017). Industry 4.0 and the current status as well as future prospects on logistics. *Computers in Industry*, *89*(Supplement C), 23–34. https://doi.org/10.1016/j.compind.2017.04.002

Hollender, M. (2010). *Collaborative Process Automation Systems*. ISA.

Huang, Z., Yu, H., Peng, Z., & Feng, Y. (2017). Planning community energy system in the

industry 4.0 era: Achievements, challenges and a potential solution. *Renewable and Sustainable Energy Reviews*, *78*(Supplement C), 710–721. https://doi.org/10.1016/j.rser.2017.04.004

Huber, A., & Weiss, A. (2017). Developing Human-Robot Interaction for an Industry 4.0 Robot: How Industry Workers Helped to Improve Remote-HRI to Physical-HRI. *Proceedings of the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, 137–138. https://doi.org/10.1145/3029798.3038346

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

Ivanov, D., Sokolov, B., & Ivanova, M. (2016). Schedule coordination in cyber-physical supply networks Industry 4.0. *IFAC-PapersOnLine*, *49*(12), 839–844. https://doi.org/10.1016/j.ifacol.2016.07.879

Jazdi, N. (2014). Cyber physical systems in the context of Industry 4.0. *2014 IEEE International Conference on Automation, Quality and Testing, Robotics*, 1–4. https://doi.org/10.1109/AQTR.2014.6857843

Jirkovský, V., Obitko, M., & Mařík, V. (2017). Understanding Data Heterogeneity in the Context of Cyber-Physical Systems Integration. *IEEE Transactions on Industrial Informatics*, *13*(2), 660–667. https://doi.org/10.1109/TII.2016.2596101

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

Kang, N., Zhao, C., Li, J., & Horst, J. A. (2016). A Hierarchical structure of key performance indicators for operation management and continuous improvement in production systems. *International Journal of Production Research*, *54*(21), 6333–6350. https://doi.org/10.1080/00207543.2015.1136082

Khan, M., Wu, X., Xu, X., & Dou, W. (2017). Big data challenges and opportunities in the hype of Industry 4.0. *2017 IEEE International Conference on Communications (ICC)*, 1–6. https://doi.org/10.1109/ICC.2017.7996801

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

Leal, G., Guédria, W., & Panetto, H. (2019). An ontology for interoperability assessment: A systemic approach. *Journal of Industrial Information Integration*, *16*, 100100. https://doi.org/10.1016/j.jii.2019.07.001

Lee, D., Choi, K., & Kim, H. (2017). Editorial: Smart Devices & Smart Spaces in Wireless Internet of Everything (Wireless-IoE). *Wireless Personal Communications*, *94*(2), 145–147. https://doi.org/10.1007/s11277-017-4103-9

Lee, J., Bagheri, B., & Kao, H.-A. (2015). A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems. *Manufacturing Letters*, *3*(Supplement C), 18–23. https://doi.org/10.1016/j.mfglet.2014.12.001

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

Li, X., Li, D., Wan, J., Vasilakos, A. V., Lai, C.-F., & Wang, S. (2017a). A review of industrial wireless networks in the context of Industry 4.0. *Wireless Networks*, *23*(1), 23–41. https://doi.org/10.1007/s11276-015-1133-7

Li, X., Li, D., Wan, J., Vasilakos, A. V., Lai, C.-F., & Wang, S. (2017b). A review of industrial wireless networks in the context of Industry 4.0. *Wireless Networks*, *23*(1), 23–41. https://doi.org/10.1007/s11276-015-1133-7

Liao, Y., Deschamps, F., Loures, E. de F. R., & Ramos, L. F. P. (2017). Past, present and future of Industry 4.0—A systematic literature review and research agenda proposal. *International Journal of Production Research*, *55*(12), 3609–3629. https://doi.org/10.1080/00207543.2017.1308576

Liu, H., Ning, H., Mu, Q., Zheng, Y., Zeng, J., Yang, L. T., Huang, R., & Ma, J. (2017). A review of the smart world. *Future Generation Computer Systems*. https://doi.org/10.1016/j.future.2017.09.010

Longo, F., Nicoletti, L., & Padovano, A. (2017). Smart operators in industry 4.0: A human-centered approach to enhance operators' capabilities and competencies within the new smart factory context. *Computers & Industrial Engineering*, *113*(Supplement C), 144–159. https://doi.org/10.1016/j.cie.2017.09.016

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

Lucas-Estañ, M. C., Raptis, T. P., Sepulcre, M., Passarella, A., Regueiro, C., & Lazaro, O. (2018). A software defined hierarchical communication and data management architecture for industry 4.0. *2018 14th Annual Conference on Wireless On-Demand Network Systems and Services (WONS)*, 37–44. https://doi.org/10.23919/WONS.2018.8311660

Mahmud, R., Kotagiri, R., & Buyya, R. (2018). *Fog computing: A taxonomy, survey and future directions*. Springer.

Malhotra, L., Agarwal, D., & Jaiswal, A. (2014). *Virtualization in Cloud Computing*.

https://doi.org/10.4172/2165-7866.1000136

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

Mezghani, E., Expósito, E., & Drira, K. (2017a). A Model-Driven Methodology for the Design of Autonomic and Cognitive IoT-Based Systems: Application to Healthcare. *IEEE Transactions on Emerging Topics in Computational Intelligence*, *1*(3), 224–234. https://doi.org/10.1109/TETCI.2017.2699218

Mezghani, E., Expósito, E., & Drira, K. (2017b). *An Autonomic Cognitive Pattern for Smart IoT-based System Manageability: Application to Comorbidity Management*. https://hal.laas.fr/hal-01651945/document

Mohamed, N., Al-Jaroodi, J., Jawhar, I., Lazarova-Molnar, S., & Mahmoud, S. (2017). SmartCityWare: A Service-Oriented Middleware for Cloud and Fog Enabled Smart City Services. *IEEE Access*, *5*, 17576–17588. https://doi.org/10.1109/ACCESS.2017.2731382

Molano, J. I. R., Lovelle, J. M. C., Montenegro, C. E., Granados, J. J. R., & Crespo, R. G. (2017). Metamodel for integration of Internet of Things, Social Networks, the Cloud and Industry 4.0. *Journal of Ambient Intelligence and Humanized Computing*, 1–15. https://doi.org/10.1007/s12652-017-0469-5

Morgan, J., & O'Donnell, G. E. (2017). Enabling a ubiquitous and cloud manufacturing foundation with field-level service-oriented architecture. *International Journal of Computer Integrated Manufacturing*, *30*(4–5), 442–458. https://doi.org/10.1080/0951192X.2015.1032355

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

Napoleone, A., Macchi, M., & Pozzetti, A. (2020). A review on the characteristics of cyber-physical systems for the future smart factories. *Journal of Manufacturing Systems*, *54*, 305–335. https://doi.org/10.1016/j.jmsy.2020.01.007

Nelles, J., Kuz, S., Mertens, A., & Schlick, C. (2016). Human-centered design of assistance systems for production planning and control. The role of the human in Industry 4.0. *IEEE International Conference on Industrial Technology (ICIT)*, 2099–2104. https://doi.org/10.1109/ICIT.2016.7475093

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

Orellana, F., & Torres, R. (2019). From legacy-based factories to smart factories level 2 according to the industry 4.0. *International Journal of Computer Integrated*

Manufacturing, *32*(4–5), 441–451. https://doi.org/10.1080/0951192X.2019.1609702

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

Panetto, H., Iung, B., Ivanov, D., Weichhart, G., & Wang, X. (2019). Challenges for the cyber-physical manufacturing enterprises of the future. *Annual Reviews in Control*, *47*, 200–213. https://doi.org/10.1016/j.arcontrol.2019.02.002

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.

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.

Penas, O., Plateaux, R., Patalano, S., & Hammadi, M. (2017). Multi-scale approach from mechatronic to Cyber-Physical Systems for the design of manufacturing systems. *Computers in Industry*, *86*, 52–69. https://doi.org/10.1016/j.compind.2016.12.001

Peng, Z., Xu, B., Gates, A. M., Cui, D., & Lin, W. (2017). A Study of a Multi-Agent Organizational Framework with Virtual Machine Clusters as the Unit of Granularity in Cloud Computing. *The Computer Journal*, *60*(7), 1032–1043. https://doi.org/10.1093/comjnl/bxw042

Perera, C., Zaslavsky, A., Christen, P., & Georgakopoulos, D. (2014). Sensing as a service model for smart cities supported by Internet of Things. *Transactions on Emerging Telecommunications Technologies*, *25*(1), 81–93. https://doi.org/10.1002/ett.2704

Pierdicca, R., Frontoni, E., Pollini, R., Trani, M., & Verdini, L. (2017). The Use of Augmented Reality Glasses for the Application in Industry 4.0. *Augmented Reality, Virtual Reality, and Computer Graphics*, *10324*, 389–401. https://doi.org/10.1007/978-3-319-60922-5_30

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

Pinto Pereira, A. (2014). *Towards robustness and self-organization of ESB-based solutions using service life-cycle management* [Master Thesis]. Polytechnic Institute of Bragança.

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

Preuveneers, D., & Ilie-Zudor, E. (2017). The intelligent industry of the future: A survey on emerging trends, research challenges and opportunities in Industry 4.0. *Journal of Ambient Intelligence and Smart Environments*, *9*(3), 287–298.

https://doi.org/10.3233/AIS-170432

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

Qin, J., Liu, Y., & Grosvenor, R. (2016). A Categorical Framework of Manufacturing for Industry 4.0 and Beyond. *Procedia CIRP*, *52*, 173–178. https://doi.org/10.1016/j.procir.2016.08.005

Qiu, M., Garg, S., Buyya, R., Yu, B., & Hu, S. (2017). Special Issue on Scalable Cyber–Physical Systems. *Journal of Parallel and Distributed Computing*, *103*(Supplement C), 1–2. https://doi.org/10.1016/j.jpdc.2017.01.025

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

Richert, A., Shehadeh, M., Müller, S., Schröder, S., & Jeschke, S. (2016). Robotic Workmates – Hybrid Human-Robot-Teams in the Industry 4.0. *International Conference on E-Learning*, 1.

Riel, A., & Flatscher, M. (2017). A Design Process Approach to Strategic Production Planning for Industry 4.0. *Systems, Software and Services Process Improvement*, 323–333. https://doi.org/10.1007/978-3-319-64218-5_27

Riggins, F., & Keskin, T. (2017, January 4). Introduction to Internet of Things: Providing Services Using Smart Devices, Wearables, and Quantified Self Minitrack. *Proceedings of the 50th Hawaii International Conference on System Sciences*. International Conference on System Sciences. https://doi.org/10.24251/HICSS.2017.166

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

Román-Ibáñez, V., Jimeno-Morenilla, A., & Pujol-López, F. A. (2018). Distributed monitoring of heterogeneous robotic cells. A proposal for the footwear industry 4.0. *International Journal of Computer Integrated Manufacturing*, *31*(12), 1205–1219. https://doi.org/10.1080/0951192X.2018.1529432

Romero, D., Wuest, T., Stahre, J., & Gorecky, D. (2017). Social Factory Architecture: Social Networking Services and Production Scenarios Through the Social Internet of Things, Services and People for the Social Operator 4.0. *Advances in Production Management Systems. The Path to Intelligent, Collaborative and Sustainable Manufacturing*, 265–273. https://doi.org/10.1007/978-3-319-66923-6_31

Roques, P. (2016, January). MBSE with the ARCADIA Method and the Capella Tool. *8th European Congress on Embedded Real Time Software and Systems (ERTS 2016)*. https://hal.archives-ouvertes.fr/hal-01258014

ROS Industrial. (2018, November 16). *ROS-Industrial*. ROS Industrial.

https://rosindustrial.org/

Rossit, D. A., Tohmé, F., & Frutos, M. (2019). Industry 4.0: Smart Scheduling. *International Journal of Production Research*, *57*(12), 3802–3813. https://doi.org/10.1080/00207543.2018.1504248

Sanchez, M., Aguilar, J., Cordero, J., & Vadiviezo, P. (2015a). A Smart Learning Environment based on Cloud Learning. *International Journal of Advanced Information Science and Technology*, *4*(7), 36–49. https://doi.org/10.15693/ijaist/2015.v4i7.36-49

Sanchez, M., Aguilar, J., Cordero, J., & Valdiviezo, P. (2015b). Basic Features of a Reflective Middleware for Intelligent Learning Environment in the Cloud (IECL). *2015 Asia-Pacific Conference on Computer Aided System Engineering*, 1–6. https://doi.org/10.1109/APCASE.2015.8

Sanchez, M., Aguilar, J., Cordero, J., Valdiviezo-Díaz, P., Barba-Guamán, L., & Chamba-Eras, L. (2016). Cloud Computing in Smart Educational Environments: Application in Learning Analytics as Service. In Á. Rocha, A. M. Correia, H. Adeli, L. P. Reis, & M. Mendonça Teixeira (Eds.), *New Advances in Information Systems and Technologies* (Vol. 444, pp. 993–1002). Springer International Publishing. https://doi.org/10.1007/978-3-319-31232-3_94

Sanchez, M., Aguilar, J., & Exposito, E. (2018a). *Industry 4.0 Survey and Challenges from a System Integration Perspective* (Tech. Rep No. 20180112; p. 12).

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

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

Sanderson, D., Chaplin, J. C., & Ratchev, S. (2018). Conceptual Framework for Ubiquitous Cyber-Physical Assembly Systems in Airframe Assembly∗∗The authors gratefully acknowledge the support provided by UK EPSRC Evolvable Assembly Systems (EP/K018205/1). *IFAC-PapersOnLine*, *51*(11), 417–422. https://doi.org/10.1016/j.ifacol.2018.08.331

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

Schwab, K. (2016). *La cuarta revolución industrial*. Penguin Random House Grupo Editorial España.

Seeger, J., Deshmukh, R. A., & Bröring, A. (2018). Dynamic IoT Choreographies—Managing Discovery, Distribution, Failure and Reconfiguration. *ArXiv:1803.03190 [Cs]*. http://arxiv.org/abs/1803.03190

Sengupta, S., Gupta, N., & Naik, V. (Advisor). (2017). *Firewall for internet of things* [Indraprastha Institute of Information Technology]. https://repository.iiitd.edu.in/xmlui/handle/123456789/587

Shafiq, M. O., Ding, Y., & Fensel, D. (2006). Bridging multi agent systems and web services: Towards interoperability between software agents and semantic web services. *Enterprise Distributed Object Computing Conference, 2006. EDOC'06. 10th IEEE International*, 85–96.

Shaikh, S. F., Ghoneim, M. T., Sevilla, G. T., Nassar, J. M., Hussain, A. M., & Hussain, M. M. (2017). Freeform Compliant CMOS Electronic Systems for Internet of Everything Applications. *IEEE Transactions on Electron Devices*, *64*(5), 1894–1905. https://doi.org/10.1109/TED.2016.2642340

Shila, D. M., Shen, W., Cheng, Y., Tian, X., & Shen, X. S. (2017). AMCloud: Toward a Secure Autonomic Mobile Ad Hoc Cloud Computing System. *IEEE Wireless Communications*, *24*(2), 74–81. https://doi.org/10.1109/MWC.2016.1500119RP

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

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

Soto, J. A. C., Tavakolizadeh, F., & Gyulai, D. (2019). An online machine learning framework for early detection of product failures in an Industry 4.0 context. *International Journal of Computer Integrated Manufacturing*, *32*(4–5), 452–465. https://doi.org/10.1080/0951192X.2019.1571238

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

Strozzi, F., Colicchia, C., Creazza, A., & Noè, C. (2017). Literature review on the 'Smart Factory' concept using bibliometric tools. *International Journal of Production Research*, *55*(22), 6572–6591. https://doi.org/10.1080/00207543.2017.1326643

Suri, K., Cuccuru, A., Cadavid, J., Gérard, S., Gaaloul, W., & Tata, S. (2017). Model-based Development of Modular Complex Systems for Accomplishing System Integration for Industry 4.0. *5th International Conference on Model-Driven Engineering and Software Development*, 487–495. https://doi.org/10.5220/0006210504870495

Syberfeldt, A., Danielsson, O., & Gustavsson, P. (2017). Augmented Reality Smart Glasses in the Smart Factory: Product Evaluation Guidelines and Review of Available Products. *IEEE Access*, *5*, 9118–9130. https://doi.org/10.1109/ACCESS.2017.2703952

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.

Tran, N.-H., Park, H.-S., Nguyen, Q.-V., & Hoang, T.-D. (2019). Development of a Smart Cyber-Physical Manufacturing System in the Industry 4.0 Context. *Applied Sciences*, *9*(16), 3325. https://doi.org/10.3390/app9163325

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

Tu, M., K. Lim, M., & Yang, M.-F. (2018). IoT-based production logistics and supply chain system – Part 2: IoT-based cyber-physical system: a framework and evaluation. *Industrial Management & Data Systems*, *118*(1), 96–125. https://doi.org/10.1108/IMDS-11-2016-0504

Vaquero, L. M., Rodero-Merino, L., Caceres, J., & Lindner, M. (2008). A Break in the Clouds: Towards a Cloud Definition. *SIGCOMM Comput. Commun. Rev.*, *39*(1), 50–55. https://doi.org/10.1145/1496091.1496100

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

Wan, J., Tang, S., Shu, Z., Li, D., Wang, S., Imran, M., & Vasilakos, A. V. (2016). Software-Defined Industrial Internet of Things in the Context of Industry 4.0. *IEEE Sensors Journal*, *16*(20), 7373–7380. https://doi.org/10.1109/JSEN.2016.2565621

Wang, S., Wan, J., Zhang, D., Li, D., & Zhang, C. (2016a). Towards smart factory for industry 4.0: A self-organized multi-agent system with big data based feedback and coordination. *Computer Networks*, *101*(Supplement C), 158–168. https://doi.org/10.1016/j.comnet.2015.12.017

Wang, S., Wan, J., Zhang, D., Li, D., & Zhang, C. (2016b). Towards smart factory for industry 4.0: A self-organized multi-agent system with big data based feedback and coordination. *Computer Networks*, *101*(Supplement C), 158–168. https://doi.org/10.1016/j.comnet.2015.12.017

Weyer, S., Schmitt, M., Ohmer, M., & Gorecky, D. (2015). Towards Industry 4.0—Standardization as the crucial challenge for highly modular, multi-vendor production systems. *IFAC (International Federation of Automatic Control) Hosting by Elsevier Ltd.*, *48*(3), 579–584. https://doi.org/10.1016/j.ifacol.2015.06.143

Xu, L. D., & Duan, L. (2019). Big data for cyber physical systems in industry 4.0: A survey. *Enterprise Information Systems*, *13*(2), 148–169. https://doi.org/10.1080/17517575.2018.1442934

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

Zanero, S. (2017). Cyber-Physical Systems. *Computer*, *50*(4), 14–16. https://doi.org/10.1109/MC.2017.105