Agent-based Service Reconfiguration for Dynamic and Evolvable Systems

Ph.D. Thesis

by

Nelson Ricardo Martins Rodrigues

A dissertation submitted to thein partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Supervisor: Eugénio da Costa Oliveira, Ph.D., University of Porto
Co-Supervisor: Paulo Leitão, Ph.D., Polytechnic Institute of Bragança

2019

October 2019

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

DEDICATION

Dedicated to my family and friends.

RESUMO

As empresas de manufactura que estão empenhadas em manter-se competitivas em mercados globais necessitam de produtos altamente personalizados, com alta qualidade e curtos prazos de entrega. Nestas circunstâncias, ser capaz de responder a mudanças constantes ao nível operacional exige reconfigurabilidade, flexibilidade, adaptabilidade e agilidade, de modo a que cada empresa se mantenha competitiva. Ao mesmo tempo, tais sistemas precisam de ser capazes de lidar com a complexidade adicional, a qual é inerente a uma reconfiguração dinâmica em sistemas evolutivos.

Desta forma, desenvolver e manipular sistemas contínuos ao mesmo tempo que as restrições em tempo real são asseguradas, pode tornar-se cada vez mais complexo, difícil de prever e, por consequência, complexo de controlar. Neste contexto, este trabalho visiona elaborar uma metodologia distribuída que suporte uma reconfiguração de serviços dinâmica e automática, num domínio em constante evolução, com foco na melhoria da utilidade do sistema.

Para tal é proposta uma arquitectura distribuída baseada em agentes, designada por ADvISER (*Reconfiguração Dinâmica de Serviços com Arquitectura de Sistemas Multi-Agente*), que promove a identificação de oportunidades de forma pro-activa. Além disso, o ADvISER recorre a propriedades autónomas (*self-**) e estratégias de reconfiguração, particularmente mecanismos auto-organizados, para devidamente endereçar novas soluções de reconfiguração, melhorando a eficiência de produção.

Cada agente ADvISER é autónomo, inteligente e enriquecido por ingredientes essenciais, como paradigma orientado a serviços, comportamentos cooperativos e autoadaptativos para explorar a flexibilidade de suporte à decisão para a reconfiguração dos serviços. A integração e análise dos resultados em cenários de automação industrial demonstra a viabilidade e os benefícios da arquitectura de sistemas multi-agente. Estes últimos tornam-se ainda mais evidentes, quanto mais dinâmico e evolucionário for o sistema.

Por fim, esta investigação aqui apresentada é suportada por várias contribuições científicas publicadas em conferências e revistas internacionais durante o período desta tese. Além disso, partes do código-fonte da arquitetura ADvISER estão incluídas em vários projetos internacionais.

ABSTRACT

Manufacturing companies that are committed to remain competitive in global markets require highly customized products with high quality and short lead times. In these circumstances, being able to respond to constant changes at the operational level requires reconfigurability, flexibility, adaptability, and agility, so that each company remains competitive. At the same time, such systems need to be able to cope with the additional complexity, which is inherent to a dynamic reconfiguration in evolutionary systems.

Accordingly, developing and manipulating continuous systems, ensuring real-time constraints simultaneously, can become increasingly complex, difficult to predict and therefore, complex to control. In this context, this work envisages the elaboration of a distributed methodology that supports dynamic and automatic reconfiguration of services, in a constantly evolving domain, focusing on improving the usefulness of the system.

To this end, a distributed agent-based architecture, called ADvISER (*A Dynamic Service Reconfiguration with a Multi-Agent System's Architecture*), is proposed that proactively identifies opportunities. In addition, ADvISER uses autonomous (*self-**) properties and reconfiguration strategies, particularly self-organized mechanisms, to properly address new reconfiguration solutions, improving production efficiency.

Each ADvISER agent is autonomous, intelligent and enriched by essential ingredients such as a service-oriented paradigm, cooperative and self-adaptive behaviors to explore the flexibility of decision support for reconfiguring services. Integrating and analyzing the results in industrial automation scenarios demonstrates the feasibility and benefits of the multi-agent system architecture. The latter becomes even more evident the more dynamic and evolutionary the system.

The work is supported by several scientific contributions published in international journals and conferences during the work on this thesis. In addition, part of the ADvISER source code is included in several international projects.

ACKNOWLEDGEMENTS

This thesis is the result of some years of research, during which I had the opportunity to work closely with many people. I would like to take this opportunity to express my gratitude to all who helped to make this thesis a reality. I would like to express my deep gratitude to my research supervisor, Professor Eugénio Oliveira, for the trust he placed in me at the beginning of the PhD's long walk.

I also would like to thank for his encouragement, precious advises and discussions, that contributed to the elaboration of this thesis, and the improvement of the final result.

My acknowledge to the Faculty of Engineering of the University of Porto, for giving me the opportunity to elaborate this research work and to my colleagues of Artificial Intelligence and Computer Science Laboratory (LIACC) at the University of Porto for the availability and support given during this work.

I am happy to contribute to several European research projects, such as GRACE (coordinated by Univ. Politecnica delle Marche and having partners SINTEF, AEA srl, Instituto Politécnico de Bragança, Whirlpool Europe srl and Siemens AG), ARUM, and PERFoRM. In this context, I am very grateful for the inspiring and fruitful discussions and all the fun I had with European researchers. Sincere thanks to Lorenzo, Giacomo, Claudio, Arnaldo and Stefano. I would like to express my gratitude to ESTiG - School of Technology and Management from the Polytechnic Institute of Bragança for the availability and support given during this work and specially to my co-supervisor Professor Paulo Leitão for granting me a research scholarship, without which this work would not have been possible.

I also wish to express my gratitude to all the colleagues from the Laboratory and Research Centre in Digitalization and Intelligent Robotics (CeDRI), for their friendship, support and inspiring discussions, that has enriched my knowledge in this research area; a special thanks to José, Flávia, Adriano, Arnaldo, Filipe, Jonas, Ana and more recently to Luis, and Marcelo.

I am deeply grateful to my parents, Eduardo and Alice, which throughout my Academic career and especially at this stage remained always supporters and motivators, for me to successfully overcome all the barriers. A special thanks to Ana and my brother Tiago for the long sessions of debating ideas, and my cousin Tiago that contributed to the success of the presented solution and to all my friends which directly or indirectly in a daily basis had to bear with my good and bad moments.

Finally, I want to leave a special note of thanks to Patrícia for the wise advice, her patience, and the many brainstorming moments that have helped me to improve and helped me stay focused. I promise I will try to spend more time with her in the future ;)

To all my thanks.

Table of Contents

List of Tables

Table **Page**

FIGURE **Page**

GLOSSARY

Please note that the terms are short and straightforward. For more detail, please follow the references into the main text.

ADvISER - Dynamic Service Reconfiguration with a Multi-Agent System Architecture, and respective Design Framework, for the regulation of manufacturing operations in a dynamic system.

ADAPTABILITY - The ability of a system to be changed by a system-internal change agent with intent.

CHANGEABILITY - The ability of a system to alter its form or operations, and consequently possibly its function, at an acceptable level of resources.

EVOLVABILITY - The ability of a design to be inherited and changed across generations (over time). (Note: this is also known as extensibility, modifiability, or plasticity).

MODULARITY - The degree to which a system is composed of modules.

PLIABILITY - The ability of a system to change, without breaking its system architecture.

RECONFIGURABILITY - The ability of a system to change its configuration (component arrangement and links).

INTEROPERABILITY - The ability of a system to effectively interact with other systems.

4IR - Fourth Industrial Revolution
7 - IR - 7 Industrial Requirements
8 - Features - 8 Features
ACL - Agent Communication Language
ACO - Ant Colony Optimization
ADvISER - *A Dynamic Service Reconfiguration with Multi-Agent Systems Architecture*
AI - Artificial Intelligence
BPEL - Business Process Execution Language
CEP - Complex Event Processing
CS - Computer Science
CPS - Cyber-Physical Systems
CPPS - Cyber-Physical Production System
DAI - Distributed Artificial Intelligence
DF - Direct Facilitator
DoF - Degrees of Freedom
DRS - Decice Reconfiguration Decision
EPS - Evolvable Production Systems
ES - Evolution Strategies
FIPA - Foundation for Intelligent Physical Agents
FT - Functional Test
FMS - Flexible Manufacturing Systems
GA - Genetic Algorithm
HtR - How to Reconfigure
I4.0 - Industry 4.0
IE - Industrial Engineering
IMS - Intelligent Manufacturing System
IoT - Internet of Things
IP - Intelligent Products
IPPS - Integration of Process Planning and Scheduling
IR - Intelligent Resources
IR - Industrial Requirement
IT - Information Technology
JADE - Java Agent Development Framework
JRMI - Java Remote Method Invocation
KPI - Key Performance Indicator
KQML - Knowledge Query and Manipulation Language
LCL - Lower Control Limit
MAS - Multi-Agent System
MES - Manufacturing Execution System

ML - Machine Learning
MLaaS - Machine Learning as a Service
MQTT - Message Queuing Telemetry Transport
MTBF - Mean Time Between Failures
OPC-UA - OPC Unified Architecture
OWL - Web Ontology Language
PA - Product Agent
PDCA - Plan–Do–Check–Act
PSS - Product Service System
PTA - Product Type Agent
QoS - Quality of Service
RA - Resource Agent
RC - Reconfiguration Cost
RDF - Resource Description Framework
RI - Reconfiguration Index
RL - Reinforcement Learning
RMS - Reconfigurable Manufacturing System
SA - Simulated Annealing
SCADA - Supervisory Control and Data Acquisition
SE - Software Engineering
SF - Service Facilitator
SOA - Service Oriented Architectures
SOC - Service-Oriented Computing
SoMAS - Service Oriented Multi-agent System
TBF - Time between Failure
TRL - Technology Readiness Level
UCL - Upper Control Limit
UDDI - Universal Description, Discovery and Integration
UML - Unified Modeling Language
WSDL - Web Service Definition Language
WSFL - Web Services Flow Language
YPA - Yellow Page Agent

CHAPTER 1

INTRODUCTION

"You should be glad that bridge fell down. I was planning to build thirteen more to that same design"

Isambard Kingdom Brunel

This chapter provides a vision of today's manufacturing sector, which is witnessing a fourth Industrial and technological revolution called Industry 4.0, with the unprecedented ability to enhance smart manufacturing automation systems with a reconfiguration mindset.

This vision offers insights that lead companies to shift away from the traditional manufacturing infrastructures, based on the rigid ISA-95 automation pyramid, and embrace emerging technologies. The technologies that have been adopted under the umbrella of Industry 4.0 enable industries to influence the necessary processes of reconfigurability, making the production system more flexible and adaptable.

Therefore, and for a better comprehension of this thesis research scope, a list of research questions is provided, along with the hypotheses of whether flexible infrastructures are capable of supporting on-the-fly reconfigurability, on the way to reach manufacturing sustainability. Then, the main expected contributions of this work are introduced, as well as the boundaries that clearly define the open points to investigate. Finally, this chapter outlines the organization of the document.

1.1 Context

In the last decades, dynamic markets have been requiring a high rate of change, with IT companies being forced to cope with it in order to stay competitive. This situation, still occurs in today's business, pushing companies from different domains, *e.g.*, manufacturing production lines, to continuously deliver high-quality services, as a way of raising their competitive position while, at the same time, facing the increasing demand for sophisticated and customized services.

However, systems that deal with constant requirements and continuous new functionalities are highly complex. For example, producing heterogeneous products in small lot sizes, new orders,

or disturbances in the system, require advanced management systems, particularly automated production systems [202] to deal with all these continuous challenges.

Although automated production systems are generally more relevant for systems that change often, they also possess benefits that could be applied in the traditional production settings. The current traditional production lines are highly rigid and organized to deliver as many products as possible in a reduced period of time [44], but they are also vulnerable to multiple and unexpected perturbations.

These systems' susceptibility becomes more evident in modern production lines than the traditional since they often embrace complex scenarios. For example, a production that requires a real-time modification of the product portfolio during the execution lifecycle, to a new family of product, requires the reconfiguration of the Industrial settings to accommodate the production of the new product, which is not a simple task. It involves software to automate a manufacturing production system [129, 215], to automatically configure the control software, while expecting to increase the reconfigurability, adaptability and also the flexibility of the system. Yet, manufacturing systems are not able to integrate such concepts due to their rigid and inflexible structure [215].

1.1.1 Evolution of the Manufacturing Paradigms

This forces a shift in the manufacturing paradigm aiming to automate[1] the different production processes. During the last decade, we have witnessed an evolution in the manufacturing production paradigms as an attempt to allow flexible and adaptable systems to become a reality. For example, Flexible Manufacturing System (FMS), Reconfigurable manufacturing system (RMS), Plug and Produce (P&P) are fundamental paradigm mechanisms to employ a changeable system for fast services/components changes like replacing modules.

1.1.2 Control System Architectures

During the last two decades, new production paradigms and various software solutions have been proposed or evolving to control manufacturing processes, aiming to optimize the production line, as an attempt to allow flexible and adaptable systems to come to life.

But the manufacturing processes have their own problems, imposing unique challenges, and the current software is not sufficient, requiring better procedures to deal with increasing complexity levels. Thus, along with the necessity of a robust system to deal with changes, we still need additional advancements in software solutions to control the production. The aim is to empower the system with a high degree of predictability and reactivity and to cope with several other challenges, such as unplanned production downtime or deviations in the systems' performance that can easily be reflected in revenue opportunities being lost.

To overcome this limitation, emerging technologies such as Multi-Agent Systems (MAS) and Service-Oriented Architecture (SOA) can bring new possibilities for the development of suitable software, capable of automatically controlling the production plan and empowering industries with a flexible and adaptable type of production system:

[1] The term automation itself descends from the Greek word *automatos* meaning self-moving, self thinking.

- SOA-based technology [121] is oftenly addressed in the Industrial manufacturing applications as a suitable approach for distributing the control system. Despite being more famous in different IT domains, in this context, service-oriented architectures deliver an important contribution, mainly providing the characteristics of a system integrator centered on loose coupling operations. This feature brings up the opportunity to achieve adaptability in the form of a set of interconnected services that are easily transformed or replaced. SOA principles not only influence the design of the reconfigurations effortlessly and as transparently as possible [72, 131], they also offer the ideal platform for runtime adaptations.
- Agent-based technology and multi-agent systems [4, 142], have been around for decades trying to revolutionize the manufacturing industry domain with multiple advantages in tackling complex problems.

Integrating these two technologies represents a significant potential. This approach considers intelligent systems enriched by essential flexible components (Services), which aims at: enabling intelligent control components, facilitating the automation of the reconfiguration of services, and at the same time simplifying the problem of additional complexity inherent in dynamic and evolutionary systems.

1.1.3 Definition of a Service Reconfigurable System

Depending on the domain, the definition of "*Service*" can describe many things. In the Industrial manufacturing domain, a generic service can be an activity provided by a hardware device, software application, like simulation tool, or even a human operator.

Curiously, also the term "*Reconfiguration*" has a strong connection to manufacturing domains mostly due to the many definitions given during the evolution of the manufacturing paradigms. As expected, for this thesis' purposes, it is adequate to give a characterization of a "*service reconfigurable system*" close to the manufacturing domain:

> *The ability of a system to change its configuration in an easy manner, without breaking its system architecture.*

This definition has some important aspects. The first one involves the concept of the "ability of a system to change, which, by definition, consists of changing/rearranging service(s). In detail, this represents a skill or competence of the system to perform the act of change, typically based on the insert, removal, update, associate or substitute actions. The second aspect considers the "*configuration*" term, oftenly this term is used in manufacturing domains to represent a particular arrangement of parts, elements or settings of a system (*e.g.*, a system, device, a computer application) or other physical or logical structure. This leads us to the fact that a *service reconfiguration* consists of the act of changing physical or logical elements (*i.e.*, services).

As stated earlier, dynamic environments are characterized by a system's reality that repeatedly imposes the need to change, which consequently inflicts a sense of resilience to change, as it can progress to something new. In this sense, a service reconfiguration system, if possible, must be performed following some basis for a smooth change. For instance, simplicity and effortless changes are two essential requirements for adopting an easy change and without difficult transitions. The fact that the changes must be in line with the new needs of the system, this directs the solution to a system with dynamic properties, that is, special mechanisms such as activation and self-motivation to be able to operate in a highly dynamic environment.

1.2 Research Problem

For a long time, good manufacturing configurations came from IT solutions dedicated to enterprise resource planning (ERP) risen from a production plan and execution level (MES). These tools create configurations resulting from the production plan and correspondent operational scheduling. They are capable of integrating several layers like ERP to the MES down to the machine control level (PLC), but it obviously involves a considerable amount of resources in terms of human resources and technology solutions, which is reflected into huge monetary costs. Naturally big companies (*e.g.*, SAP) can afford better solutions focused on larger business applications, however, this does not represent the general Industrial scenario in SME [55].

Additionally, this all-in-one type of applications continues to follow the traditional manufacturing control architectures, which do not demonstrate the flexibility or adaptability necessary for the system reconfiguration evolution at the production control level. Centralized approaches are capable of performing good optimization of the resources configuration, but their main disadvantage resides on the inherent rigidity of their architecture, which demands considerable engineering efforts for the service's modification (*e.g.*, hardware or software) [119]. Moreover, centralizing the decisions may not only create a single point of failure vulnerability but also increase the computational overhead [189], which may halt the entire manufacturing system. Heterarchical-like control approaches could present a good response to handle with distributed changes and overcome this limitation but, on the other hand, decentralized decisions, and the distributed knowledge about the system may degrade the global optimization and compromise future modifications.

In these circumstances, the challenge is to develop software that can automate the manufacturing control systems and regulate production. This software should be both autonomous and intelligent since these are essential ingredients of the reconfiguration process. Combined they allow the occurrence of reactive reconfigurations that qualify the system to deal with the disturbing event and to build knowledge capable of predicting the system performance when facing deviations. It is also expected that such software could increase the reconfigurability, adaptability, and flexibility of the system.

Although literature works present several approaches on how to manage changes and disruptions that occur during the process execution, these high-level solutions are mainly focused on collecting and analysing big amounts of data, and composed by advanced algorithms that try to identify maintenance tasks, quality deviations, optimizing the allocations of services/resources, among other functionalities. In fact, having a flexible system that is capable of adapting or embracing adaptive reconfiguration actions is a midway condition for the system to evolve into a better state. However, it does not warrant an ideal service recovery, nor optimizes the services' degradation or even maximizes the performance of the system.

This unmet situation provides opportunities to explore methodologies of service reconfiguration engineering. Another area of research targets the development of methodologies that facilitate and assist engineers during the reconfiguration design process. So far, automated and dynamic reconfiguration processes are still far from being accomplished. This can be partially explained by the lack of design guidelines/principles capable of orientating the reconfiguration process and, consequently, regulating the evolution of adaptive systems in an effective way. This gap will be at the basis of this thesis' research questions, leading to the subsequent exploration of related

emerging questions such as "*how should the system be reconfigured?*", "*what can be changed?*" or "*why and when should the system evolve?*".

This justifies our interest in comprehending the reconfiguration principles that are important when designing an effective framework for service reconfiguration. That includes the development of mechanisms with intelligent capabilities, to easily support reconfigurable systems and, therefore, allow the efficient regulation of the manufacturing operations.

1.3 Research Questions

The reconfiguration process typically uses two distinct reconfiguration strategies: adaptative and evolutive. The former allows the system to recover from disruptions, while the latter relies on adaptative procedures, equipping the system with the flexibility to consider new requirements that were not initially thought.

From this generic point of view, the question of how reconfiguration studies can be applied to the manufacturing field, thus supporting Industrial production, stands out. However, understanding and establishing *how* and *when* the dynamic service reconfigurations should be addressed, in order to improve the performance of the manufacturing reconfigurable system, contains its own panoply of new questions and challenges.

Considering this, and the lack of design guidelines/ principles capable of orientating and efficiently regulating the reconfiguration process tackled in section 1.2, we established as key research questions for this thesis the following:

> **RQ 1:** What are the main engineering design guidelines/principles that should be considered in reconfigurable systems, particularly to support an effective decision-making process that would smooth subsequent reconfigurations?
>
> **RQ 2:** How to engineer a flexible and reconfigurable architecture for exploiting adaptability algorithms, capable of identifying reconfiguration opportunities and to provide a practical step-by-step reconfiguration process?

These key research questions raise several discussion points, whose debate would represent a significant contribution to the field. Considering them, the hypothesis of this work is:

> *If each computational entity of a system could regulate itself at key moments, then it would be possible to solve reconfiguration problems and promote improving opportunities in a more flexible, reliable, dynamic and intelligent manner.*

In order to limit this hypothesis' range, particularly in the smart manufacturing field, the following sub-hypotheses were derived from the previous one established:

I. If distributed reconfiguration mechanisms are used, then the integration of services and agent-based technology will improve production control performance.

II. Integrating analytical models and AI algorithms into intelligent agents, to model reconfiguration opportunities, will result in significant improvements in the Industrial environment.

III. If collaborative service reconfiguration procedures take advantage of responsive and decentralized control decisions, it will enable them to build strategic profiles at runtime, in a flexible, autonomous and distributed environment.

IV. Both current and old service reconfiguration procedures can be used to build better strategic profiles, model triggering actions and limit the future generated strategies that need to be evaluated and addressed.

V. If the above-mentioned sub-hypothesis and the proposed technology prove to be true and useful, it will be possible to build a system capable of regulating the Industrial settings in scenarios of change (*i.e.*, unexpected disturbances like service failures or new production orders), improving the manufacturing system's performance.

Given the wide range of possible applications of this approach, it is crucial to clearly define and limit the focus and boundaries of the previous hypothesis. Therefore, in the context of this work, all the conclusions were limited to the service reconfiguration on the automation field.

1.4 Contributions & Limitations

This thesis aims at:

- Studying how reconfiguration can be used to regulate the evolution of an adaptive system (*e.g.*, manufacturing scenario).
- Reviewing and comprehending the side effects of reconfiguration in Industrial settings acquired during the system's runtime, in a way that shapes future generation strategies.

The focus of this work falls mainly on service configurations based on the fact that the traditional manufacturing approaches do not tackle the reconfiguration process adequately in the Industrial settings, which may cause the growth of inefficient and costly modifications.

In this way, the above-suggested contributions go beyond the existing state-of-the-art methodologies for dynamic and service reconfiguration, some particular questions (why, when, how, what) are also explored for designing an effective, flexible and reconfigurable system.

The developed work, in line with the search for answers and presented objectives, proposes a proposes a multi-agent software system architecture with a specific methodology in the direction of the service reconfiguration, designated by ADvISER (*A Dynamic Service Reconfiguration with a Multi-Agent Systems Architecture*), which addresses the following contributions points:

I. To explore service reconfiguration issues, drawn from experiences.

II. To provide a methodology capable of continually monitor distributed data, analysing the relevant data to diagnose and find out what happened, and propose, based on the prescriptive analytics, the automatic reconfiguration actions with as minimal effort as possible. The methodology must address the following reconfiguration challenges:

    1. Evaluating service reconfigurations efforts (costs).
    2. Analysing triggering policies for service reconfiguration.

III. To automatically associate reconfiguration opportunities across ways, through service invocation feedbacks that may be simple reactions to events or resulting from predictive models.

IV. Define a decentralized and intelligent adaptation scheme, based on multi-agent systems, to leverage distributed and variable processes to reinforce and evolve. The ADvISER's agents perform service reconfiguration procedures that are supported by the two previous points, *i.e.*, service analytics and continuous feedback behaviour to improve the decision making over the consumed and provided services.

V. Test the methodology through prototyping and demonstration of a flexible and realistic manufacturing case study, assessing the feasibility of analysing a wide range of manufacturing events that the methodology must evaluate at runtime.

For a correct validation of several contributions, it is necessary to limit the scope of the work. In this way, some assumptions are made to reduce the domain and limit the focus of the work:

I. The concept of adaptation is directed solely to entities capable of being reconfigured (*i.e.*, machines and robots with reconfiguration capability).

II. The trust and reputation will be used as useful mechanisms to ensure the correctness and resilience of the dynamic service system reconfiguration.

III. Learning is considered as the improvement of the reasoning process associated with the composition of the tasks and the evolution of the self-organization, meaning a proper evolution of the processes according to past events.

In summary, this work will consider techniques in a specific and limited scope, since it is impossible to cover all possible situations in this thesis. The innovative work concerns the agglomeration of several techniques, and technologies. The objective is to join them correctly to design systems that dynamically reconfigure the services they offer, to provide better and new services.

Most of these results appeared in Appendix A that lists the most important and significant scientific contributions published in **Q1**[2] labelled international journals and conferences during the work period for this thesis. Some of the contributions also reflect the participation in some international projects, directly related to this work domain, during these years. For instance, the ADvISER source-code is included in several international projects.

1.5 Research Methodology

The type of investigation carried out in this work follows the classical scientific research type methodology. This research process implies the refinement of the research questions and hypothesis all along the research process. The multi-disciplinary domain of this thesis is limited to a few examples of prototypic implementations that permit by means of experiments to verify the feasibility of the research questions and hypothesis.

The qualitative and quantitative evaluations are performed through the continuous analysis and measurement of the case study system. The continuous monitoring allows detecting both behavioural and quality requirements violations. This assessment will allow understanding what

[2] According to the *Scimago Journal Ranking*.

the system capabilities, and how well the ADvISER proposal performs, using **quantitative** performance analysis over non-functional requirements to comprehend how the system should work, and **qualitative** analysis of functional properties to realize what the system should do.

1.6 Thesis Outline

This document is structured to address each one of the objectives mentioned above in a systematic way.

Chapter I: In this chapter, it is offered an initial introduction to clarify the scope of the research, research challenges and objectives of this thesis.

Chapter II: Gives an overview of the manufacturing service reconfiguration domain that is best suited for dynamic and automatic reconfiguration while identifying their benefits and limitations as well as potential improvements. This chapter also describes the relevant reconfiguration requirements and the various necessary questions in order to design a flexible and reconfigurable system. This makes possible to precisely demonstrate the current limitations and possible gaps and open issues that deserve to be searched.

Chapter III: Describes the implementation of the ADvISER prototype architecture to validate the proposed reconfiguration methods. Initially, an overview of the development of a platform for the reconfiguration of services is described. Finally, the implementation of the prototype is detailed in order to create a modular, adaptive and decentralized infrastructure, along with collaborative development specifications.

Chapter IV: Gives an overview of models to be used on the ADvISER architecture particularly by each entity, focusing on the service reconfiguration mechanisms to work at runtime and offline. The conclusion phase offers in these last chapters the final result of the proposed thesis and its final discussion.

Chapter V: This chapter is devoted to the practical description of the development and validation of the proposed methods. The results of using the ADvISER architecture in scenarios of realistic production systems are also illustrated in this chapter.

Chapter VI: It discusses the conclusions reached during the development process, pointing out the contributions that are linked to the research questions and hypotheses of this work.

CHAPTER 2

THEORETICAL BACKGROUND

"Change might not be fast and it isn't always easy.
But with time and effort, almost any habit can be reshaped."

The Power of Habit

This chapter introduces the reader to some supporting concepts and relevant concepts about reconfiguration. Specifically, requirements and common questions usually used in this domain. Then, it gives an overview of some technologies enablers and the paradigms under consideration by the state-of-the-art on industrial applications for designing flexible and reconfigurable systems.

2.1 Basic Theory and Concepts

This section overviews the concepts that will support further discussions along with this document. In particular, in this work, the vision of "reconfiguration" is used in the broadest sense to refer to the several industrial phases required to evolve and modernize the existing systems of the factories into dynamic systems of factories-of-the-future to maintain the system and improve business agility.

2.1.1 Requirements for Reconfigurable Manufacturing Systems

As reported by Koren et al., [52, 89], the design requirements for achieving a reconfigurable manufacturing system require some essential core features [17, 90, 188] to keep the modification actions within acceptable limits and at the same time in operation.

- *Modularity*, the compartmentalization of operational functions into units that can be manipulated among alternate production schemes for optimal arrangements;
- *Integrability*, the ability to connect modules rapidly and precisely by a set of mechanical, informative and control interfaces facilitating integration and communication;
- *Diagnosability*, the system's ability to self-read its current state to detect and diagnose the root causes of the product's defects, quickly correcting them;

- *Convertibility*, the ability to easily transform the functionality of existing systems and machines to suit new production and market requirements;
- *Customization*, the system, and machine flexibility limited to a single product family, thereby obtaining customized flexibility;
- *Scalability*, the ability to modify the production capacity easily, by adding or removing resources and changing the system's components.

In an effort to gain a deeper understanding, Ross et al., [178] proposed a semantic concept named "Ilities" to define a coherent set of system properties and the relations among them. Table 2.1, illustrates a list of these definitions.

Table 2.1: List of Ilities with the related definitions (based on [39]).

Ility Name	**Description ("ability of a system…")**
adaptability	to be changed by a system-internal change agent with intent
agility	to change in a timely fashion
changeability	to alter its operations or form, and consequently possibly its function, at an acceptable level of resources
evolvability	design to be inherited and changed across generations (over time)
extensibility	to accommodate new features after design
flexibility	to be changed by a system-external change agent with intent
interoperability	to effectively interact with other systems
modifiability	to change the current set of specified system parameters
modularity	degree to which a system is composed of modules (not an ability-type ility)
reconfigurability	to change its component arrangement and links reversibly
robustness	to maintain its level and/or set of specified parameters in the context of changing system external and internal forces
scalability	to change the current level of a specified system parameter
survivability	to minimize the impact of a finite duration disturbance on value delivery
value robustness	to maintain value delivery in spite of changes in needs or context
versatility	to satisfy diverse needs for the system without having to change form (measure of latent value)

The proposed properties of engineering systems are not a universal solution, as mentioned by [39]. Some of these concepts will be revised during the thesis. As an example the difference between "flexibility" and "adaptability" is whether the change agent is external or internal to the system's boundary, respectively.

2.1.2 Automation and Control

As the paradigms of manufacture, also the architectures of control systems have been the target of evolutionary stages. Figure 2.1 illustrates the nature of a control system and its characteristics. As stated by in [73], a clear example of this evolution has focused on the control architectures used in manufacturing systems. This control has been discussed over the years about its advantages of being centralized and distributed.

Several examples are found across the literature concerning the evolution of the traditional model

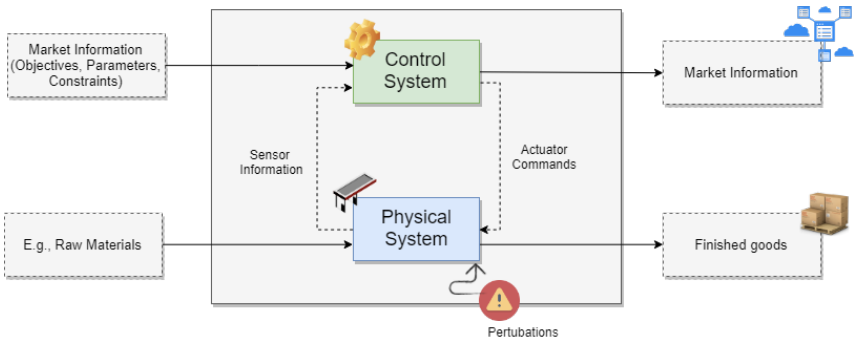

Figure 2.1: Block-diagram of manufacturing control (Source: based on [73]).

to the modern one. A simple way to identify the differences between traditional and modern automation is exemplified in the focus on which both work.

Traditional Automation is mentioned several times due to support at the control and supervision level, which is used to ensure the plans that have been defined at the level above. Typically, the traditional approach type is designed for specific products with anticipated demands. However, this type of approach has some disadvantages when it is necessary to deal with the decision to withstand disturbances or urgent orders [140].

Modern Automation, on the other hand, is designed to be ready to change throughout the levels, which means it can bring together top-level decisions from higher levels with efficiency and resource scheduling. Modern automation helps manufacturing companies to remain competitive since they have to predict potential production modifications and costumers needs to anticipate the production plan and reduce unplanned decisions. In this particular case, schedulers with forecasting mechanisms are playing an important role in permitting to optimize the production, planning decisions within predictive domains. In contrast, dynamic domains demand a constant rescheduling time, which sometimes is impractical and compromise the ability to compete in a constantly changing marketplace. Researchers suggest tackling this problem by considering answers with more flexibility, robustness and reconfigurable, which permits that whenever there are small variations, the resources be able to reconfigure dynamically, reacting promptly to unexpected events, to adapt to the desired production.

Centralized versus Decentralized

Often we find control systems based on centralized and hierarchical structures. That in itself is not harmful in static environments, because it has excellent characteristics in terms of robustness, global optimization, and predictability. In contrast, dynamic domains demand a constant rescheduling time, which sometimes is impractical and compromise the ability to compete in a constantly changing marketplace.

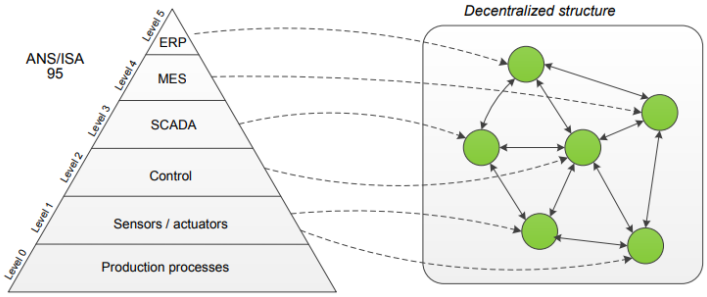

Figure 2.2: Moving from ISA-95 towards a decentralized structure (Source: based on [67]).

Distributed control architectures with non-hierarchical modules linked together through different communication systems can be more agile and responsive [50] and therefore more suitable for flexible and reconfigurable systems. Besides the local decisions, cooperation at the field level permits to implement a collaborative automation paradigm, by breaking the traditional ISA-95 automation pyramid[1], into a distributed intelligence, *e.g.*, distributed automation devices that permit to change and reorganize the components repeatedly, as illustrated in Fig. 2.2. As a result, traditional production systems are being adjusted to comply with the new requirements of intelligent and real-time reconfigurable solutions. This trend is becoming more explicit in the developments of flexible production processes in Industry 4.0 [174]. Nevertheless, it raises the complexity of the issues. On one hand, the motivation for developing such systems is to handle with the inability to deal with a complex system that is susceptible to continuous change. On the other hand, the process of designing and developing such systems efficiently leads to a more complex system.

2.1.3 Handle Complexity

Although the acceptance of new technologies might promise greater flexibility and potential to beneficially update manufacturing production lines, their adoption can also add factors capable of aggravating complexity. Although technology modernization aims at reducing complexity in uncertain and changeable systems, it can also paradoxically complicate the global infrastructure that needs to be addressed, *e.g.*, by dealing with a increased number of variables, managing more data and integrating different technological solutions. This might require rethinking the architecture of complex systems. As stated by Somashekara *et al.*, when developing complex systems the primary goal in to hide their complexity or to create the illusion of simplicity [192]. Based on this, many authors describe two fundamental concepts to manage complexity:

- *Decomposition:* the ability of a system to be separated into essential elements (decomposability).
- *Abstraction/Encapsulation:* leads to the encapsulation of the parts, facilitates the bounding of the information according to one function/process.

[1] www.isa95.org

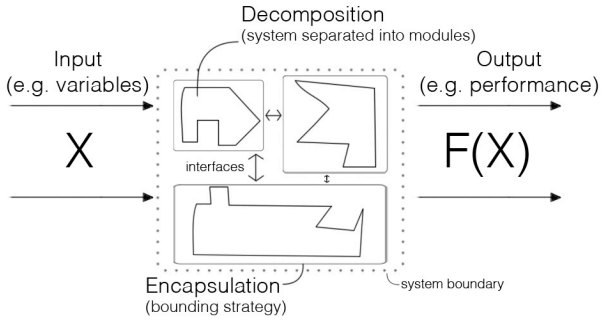

Figure 2.3: The two strategies to handle complexity: decomposition and encapsulation (Source: based on [71]).

One of the goals when designing a complex system, based on decomposition and encapsulation, is to decrease complexity by not including background details. Essentially by dividing the solution into independent, smaller, and simple functions, is possible to hide unnecessary details, following the common technique suggested by Dijkstra "*The technique of mastering complexity has been known since ancient times: divide et impera (divide and rule)*" [43]. This principle is quite popular and has influenced many engineering systems, *e.g.*, most of the complex manufacturing systems have been constructed in the form of a hierarchy that can take advantage of these principles. This approach of decomposing and encapsulating the information is aligned with a possible response to the first sub-hypothesis previously specified.

If distributed reconfiguration mechanisms are used, then the integration of services and agent-based technology will improve production control performance.

Engineering a reconfiguration mechanism capable of managing adaptions over time, on an already complex system, will require additional efforts. Unfortunately, it is inevitable to add some complexity when introducing any solution that will help the disruption of the system, so it is the key idea to introduce as little as possible. Therefore, a reasonable strategy for reducing and not improving the complexity is to adopt the decomposition and encapsulation concepts previous stated.

Handle Complexity with SOA and MAS

Service-oriented architecture is a widely used paradigm for the development of a complex system. It has a broad range of principles that can help reduce complexity. By embracing an (SOA)-based approach, we can benefit from many principles like reusability and simplicity, but more important is the **encapsulation** capability, which can be implemented with web services. Technically, SOA architecture provides a design model centered in services rather than applications that works well from the **decomposition** perspective by dividing into small, independent pieces of software in a distributed system architecture. Another excellent way to decompose a system to reduce complexity is by following

Agent Technology, particularly with many entities (*i.e.*, MAS). This technology is oftenly mentioned in order to overcome the "*divide and conquer*" problems. The idea behind this new paradigm, like SOA, is to divide and distribute the main goal into system's sub-tasks among distinct agents so that the system's overall performance would improve.

To technically address a complex system, the adoption of these technologies is aligned with the design interests of ADvISER. That is, in order to offer a system of reconfiguration without adding more complexity. SOA and MAS are two excellent approaches with exciting differences that will be discussed in detail in the forthcoming sections.

2.1.4 An Evolution from Traditional to Modern Automation

As previously described, SOA and MAS are two closely associated paradigms, which are wildly applied for easy penetration of intelligence into the automation environment taking into account the system complexity. During the last years seems that sophisticated AI tools and ML algorithms are inevitable to distribute the intelligence and disrupt the complexity in the management and control of automation systems (*e.g.*, supply chains). To penetrate such algorithms and tools is often envisioned modern architectures and technologies such as (SOA)-based frameworks to help to design advanced systems. This permit to engineering a system that gathers not only modular and flexible abilities but also to integrate a distributed cognitive system.

The successful evolution from traditional infrastructure systems to the next generation of Industrial systems requires designing a very precise migration planning. As with a typical IT migration process (*e.g.*, server, platform, etc.), all steps should be evaluated and detailed in order to minimize risk and ensure efficiency in the execution of migration steps to run as expected. To reduce risk and avoid errors during the migration process, a good practice is to follow all types of standards.

Transformation to a Modern Factory Using Flexibility and Adaptability

This is a necessary feature for a system that wants to evolve as advocated by ROSS et al., [179]. Through physical and logical change, Gonçalves [73] explores this topic, in particular the concept of "change" as an ability to modify over time. In Industry 4.0. Flexibility and adaptability are often considered to be the predominant factors of change, but they are essentially different. For a better understanding, Fig. 2.4 illustrates a relationship between them. By definition, flexibility represents the ability to react to change within a predetermined scope of requirements [12]. While adaptability means the ability to change internally. During the introduction of Industry 4.0, new requirements will naturally emerge with the change process, so the process in the automation environment needs to be able to change and modify.

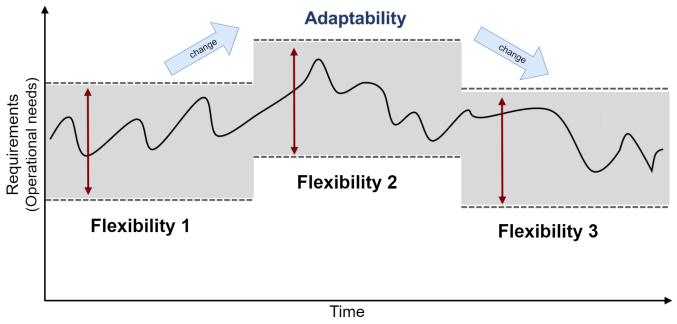

Figure 2.4: Distinction flexibility versus adaptability (Source: adapted from [12]).

In an automation environment, this migration process requires to keep in mind the various legacy systems that are installed following the specifications of the ISA-95 enterprise reference architecture. The traditional pyramid view of automation is built on different levels that will be expressed into services. So, the next generation of industrial systems is inspired by SOA integration principles that are based on consistent standards to make integration successful and, more important, flexible and easy to evolve over time.

In the following sections (2.3) some explanations will be given to how standardization plays a key role in supporting smooth industrial migration, by easily interfacing with existing systems, "plug-and-play" systems and algorithms, and adapting their behaviour and relationships on-the-fly.

2.1.5 Reconfiguration Concept

As already introduced in the previous paragraph, the term reconfiguration suggests an ability to modify an existing system's configuration [92].

2.1.6 Static vs. Dynamic Reconfiguration Process

In general, across the distinct domains, the reconfiguration concept is presented in two basic forms: *static* and *dynamic* reconfiguration.

2.1.6.1 Static Reconfiguration

Typically, this type of reconfiguration is the simplest one, because it assumes a series of fundamental characteristics that are quite straightforward (see Fig. 2.5):

- Normally, the necessary changes do not follow a (hard) real-time constraint.
- Operations may be stopped in order to load new configurations again.
- Support of the human expertise input.

As illustrated in Fig. 2.5, the static reconfiguration focus on two distinct moments of action, *i.e.*, with a stopped and running system. In this context, while the reconfiguration is taking

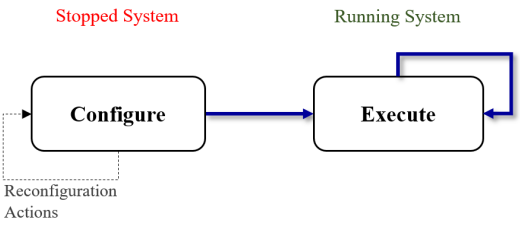

Figure 2.5: Naive formulation of static reconfiguration conceptual view.

place, the system is dedicated only to it, and no other processes are running. This type of reconfiguration is oftenly referred to as "*compiled time*" or "*offline reconfiguration*" [204]. In this situation, the resources are on the "*offline*" mode until the end of the reconfiguration process and totally available to be managed by the human during the design phase. A good example of this type of reconfiguration is mentioned by CAD tools [204] and [135], where the human stop the system in order to improve the application and start it again. This situation is useful in the sense that it allows the human plan the best configurations for the future without worry with the time constraints or the state of the system (which is static). However, this is only possible if it can monitor the state of the running system.

2.1.6.2 Dynamic Reconfiguration

As mentioned, the static reconfiguration in order to be reconfigured requires an interruption on the resources that are in execution, in another hand, the dynamic reconfiguration allows the reconfiguration and execution to proceed at the same time, this is often known as "*runtime*" and "*online*" reconfiguration (see Fig. 2.6) [80].

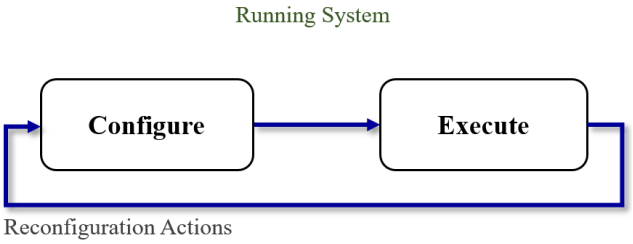

Figure 2.6: Naive formulation of dynamic reconfiguration conceptual.

Continue using the application while it is reconfiguring makes these applications more powerful and more flexible. Actually, runtime flexibility is one of the main reasons to address this type of solution. But in contrast it requires special attention to other characteristics:

- Satisfying the real-time constraint while is running.
- Continuous feedback of the system.

Manipulate a continuous system while assuring that the real-time restrictions are met might become quite a complex process. Within this context, it is necessary mechanisms that support the additional computational that is inherent to a dynamic reconfiguration approach. Create applications that are dynamic reconfigurable introduces additional design challenges, it implies to design a flexible system capable of being extended at runtime to a wide range of applications and remain flexible despite the fact that needs to tackle the extra workload. Moreover, issues like reconfiguration coordination, overhead, reconfiguration time, a sequence of reconfigurations are something that must be investigated in the next chapters to maintain the system flexible, adaptable but at the same time keep the system stability and integrity [227].

2.1.7 Service Reconfiguration Systems

Service reconfiguration is an adaptive process that happens over time aiming to modify one or more services that compose the service composition to fit the requirements or new upcoming situations. In this section, besides pointing out how the service reconfiguration process happens (*e.g.*, static or dynamic) other conditions are investigated and addressed individually.

Defining Service Reconfiguration

Depending on the domain, the definition of "*Service*" can describe many things. In the industrial manufacturing domain, a generic service can be an activity provided by a hardware device, software application, like simulation tool, or even a human operator. Curiously, also the term "*Reconfiguration*" has a strong connection to manufacturing domains mostly due to the many definitions given during the evolution of the manufacturing paradigms. Based on the several existing manufacturing reconfiguration definitions, such as in [16, 58, 188] the most common is stated by [95]:

> "*designed at the outset for rapid change in structure, as well as in hardware and software components, in order to quickly adjust production capacity and functionality within a part family in response to sudden changes in the market or regulatory requirement*".

Also in the context of manufacturing service reconfiguration can be described as [170]:

> "*Online manufacturing reconfiguration is the ability to dynamically, proactively and repeatedly adapt the hardware and software components, to provide better services in a cost-effective way*".

In detail, this represents a skill or competence of the system to perform the act of change, typically based on the insert, removal, update, associate or substitute actions. The second aspect considers the "configuration" term, oftenly this term is used in the manufacturing

domain to represent a particular arrangement of parts, elements or settings of a system (*e.g.*, a system, device, a computer application) or other physical or logical structure. This leads us to the fact that a service reconfiguration consists of the act of changing physical or logical elements (*i.e.*, services).

As stated earlier, dynamic environments are characterized by a system's reality that repeatedly imposes the need to change, which consequently inflicts a sense of resilience to change, as it can progress to something new. In this sense, a service reconfiguration system, if possible, must be performed following some basis for a smooth change. For instance, simplicity and effortless changes are two essential requirements for adopting an easy change and without difficult transitions. The fact that the changes must be in line with the new needs of the system, this directs the solution to a system with dynamic properties, that is, special mechanisms such as activation and self-motivation to be able to operate in a highly dynamic environment.

Throughout this work, flexibility is considered an important feature to achieve the principles of reconfigurability. Therefore, in addition to reconfigurability, flexibility has also become necessary criteria tools.

2.1.8 Supporting Technology Summary

As mentioned in the previous chapter, intelligent production is supported by various concepts and mechanisms, such as *Services*, *Agents*, *CPS*, and *Reconfigurations*, which have a special emphasis: reconfiguration may represent the ability to provide flexibility to the system itself. In the best of scenarios even without human intervention and without the need for the system. These are essential ingredients for an adaptive and agile manufacturing type. Then we will explore and analyze the existing work within this concept of agile manufacturing and considering the mechanisms already presented.

2.2 Requirements for Reconfigurable Systems

2.2.1 Reconfiguration Methodology

It makes no sense to talk about reconfiguration without firstly specifying and defining precisely the reconfiguration scope of the current work. This work intends to contribute to the advancement of the state-of-the-art methodologies regarding the reconfiguration process in a manufacturing environment. Therefore, in this chapter, we start by introducing the reconfiguration essentials and continue by introducing relevant questions.

2.2.2 The 5 W's and 1 H of the Reconfiguration Requirements

The requirements to execute suitable service reconfigurations will be herein reviewed and structured. This is a process that represents, by itself, an original guideline-structured contribution of this thesis.

Performing service reconfiguration traditionally means using proper services to maintain or eliminate the deterioration of the system's performance [181]. Besides understanding what the characteristics of the system are, it is indispensable to identify *"Why"* and *"What"* to modify to have a meaningful impact, and establish *When* to initiate such reconfiguration. These modifications require an initial understanding of *Who* will be the actor/entity responsible for applying the modifications, and *"How"* to reconfigure [14, 78] to a different configuration considers the service reconfiguration context. Figure 2.7 illustrates this case in greater detail.

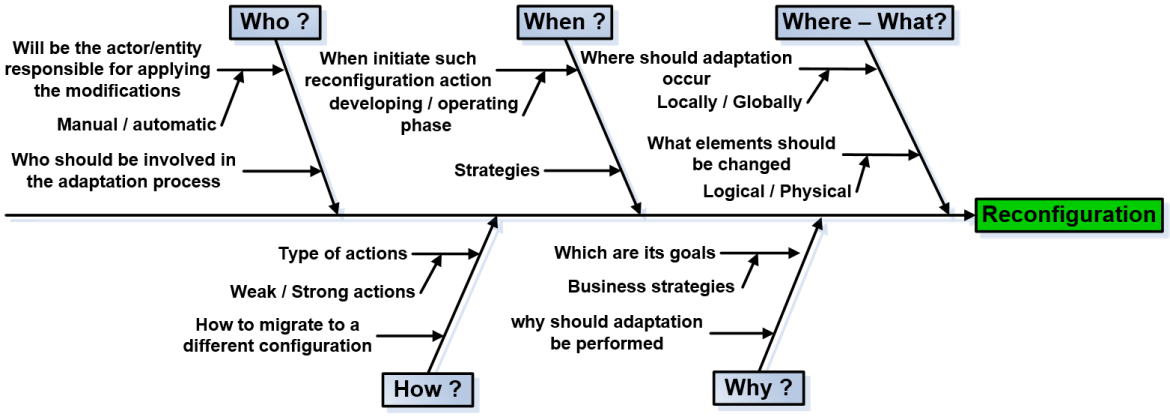

Figure 2.7: Fishbone diagram of Manufacturing Service Reconfiguration Requirements.

The success of the service reconfiguration depends on the accuracy of each spine following requirements. Each of the remaining subsections sets out the reconfiguration requisites to include within our framework, based on the fishbone diagram (see Fig. 2.7), which illustrates the requirements and the sub-requirements for the proposed reconfiguration.

2.2.3 Why Use Service Reconfiguration

This section presents a variety of reasons that support the need for change, that is, situations in which the system shows signs of inefficiency and, therefore, an opportunity for improvement is created.

There are several applications in the real-time automation domain, where changes are inevitable to maintain the system. Some of these changes are pre-defined and can be well planned, other events (*e.g.*, incidents) create inefficiencies due to their unpredictability. Ideally, incident management is one of the most common and intuitive moments to identify improvements in the system. In other words, whenever there is an identified incident, there is an opportunity to recover and also to take action capable of improving the process. In this way, it is easy to understand why recovery strategy is often mentioned in *Service Reconfiguration*, but this is not the only reason. Many other policies are dealing with the reconfiguration to promote improvements and lead to better production efficiency.

In this sub-section, distinctive and fundamental reasons to justify the "*why to reconfigure*" are explored in Table 2.2.

Table 2.2: Business strategies for performing Service Reconfiguration, resuming all the strategies to improve the system efficiency based on different methodologies.

Strategy	**Description**
Corrective	adaptation aims to optimize the faulty behavior or to replace it with a new version that provides the same functionality (for example a particular active service becomes unavailable)
Perfective	intends to improve the application even if it runs correctly
Preventive	aims to prevent upcoming faults and errors before they occur
Extending	allows to extend the application and add functionalities in response to the variations of the market, which affect the production operations

Table 2.2, resumes all strategies to improve the system efficiency based on different methodologies.

2.2.4 Reconfiguration Triggers

This sub-section presents a variety of opportunities to start a change. Whatever the purpose and methods of reconfiguration, there are several moments to execute a system adaptation. Table 2.3 rounds up into four strategies:

Table 2.3: Types of Service Reconfiguration.

Strategy	**Description**
Proactive	constantly assess the system and predicts adaptation needs
Reactive	handle faults and recover from problems reported during execution
Periodic	from time to time checks the health of the system to discover unseen opportunities
Post-mortem	to modify (or evolve) the system at design time or when stopped

According to Table 2.3, proactive and periodic policies have a dynamic context since they can be triggered at variable times, while reactive or *post-mortem* policies may or may not be automatically and/or manually triggered by human intervention.

The strategies are applied in two chronologically different moments, namely *Design-time* and *Runtime*. Independently of that, automatic triggering should preferably occur without human intervention in both cases. In contrast, the design-time considers the initial period, where the manual intervention is appropriate for the factory settings, such as before starting a new production order, where the system is adjusted to accommodate the new product production.

On the other hand, the runtime represents modifications after the production has started. The dynamic service reconfigurations assumed in this phase are performed *online* (no

need to stop, reprogram and restart the component again), with these logical modifications providing dynamism to the reconfiguration of the manufacture. In opposite, *offline* modifications are often used to make difficult reconfigurations that require to shut down the workstation during the reconfiguration execution.

2.2.5 Where/What to Change During the Service Reconfiguration

In the service manufacturing domain, reconfigurable and flexibility actions are needed to guarantee the integration of the changes in a heterogeneous way. At this point, it is necessary to summarize the functional requirements that will help to decide "*What to reconfigure*", which implies understanding the reason(s) behind the need to reconfigure. SOA represents a key technology to create an integration within and between different levels of the ISA-95 architecture [84], these levels create a pyramid of layers with varying levels of change (see Fig. 2.8) where reconfiguration can be applied.

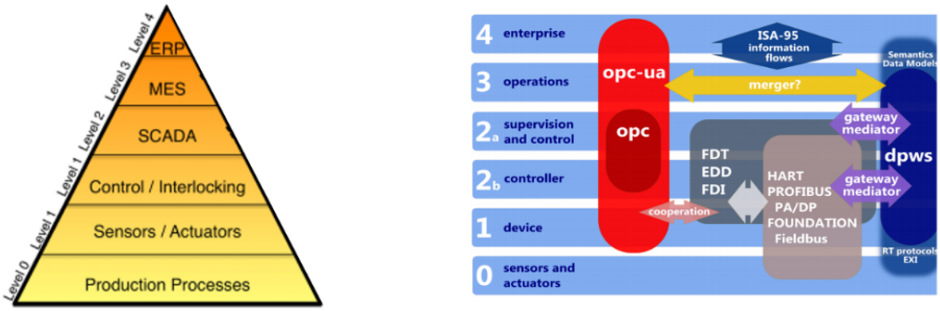

Figure 2.8: ISA-95 Layers and relevant technologies (adapted from [84]).

Current technologies provide the necessary tools to reconfigure different levels of change (such as, software programs, physical connectors), which enables interoperability throughout the system. SOA-based interactions, OPC-UA, DPWS or socket connection support a cross-layer integration and direct communication between different layers of the ISA-95, which allows integrating various types of devices [84].

Along with the different levels where service reconfiguration can be carried out, there are two main types where reconfiguration can be categorized [224], namely hard-reconfiguration and soft-reconfiguration [89]:

1. Physical (hard) reconfiguration - ability to displace and reassign the entities, *e.g.*, displace the production resources from their physical location.
2. Logical (soft) reconfiguration - reconfigure the system without physical movement, *e.g.*, planning and allocation processes.

The reconfiguration categories can be associated with the granularity of the factory level [214]. In other words, the reconfiguration ranges from a basic component to the whole factory, logically and physically. Depending on the level to reconfigure it is essential to consider different reconfiguration factors (*i.e.*, effort, profit, risk and time) independently

of being logical or physical.

Processes functionalities can be modelled as services. Thus, in this context the logical process of service reconfiguration assumes an interesting role in supporting dynamic reconfigurations. This SOA-based view is shared among many control system architectures that have been recently developed. For example, CPS is a good initiative that is becoming increasingly popular for the development of flexible production processes within Industry 4.0 [174]. Figure 2.9 illustrates an example of how migration to a future automation engineering can be achieved, which is mainly based on a service-oriented approach.

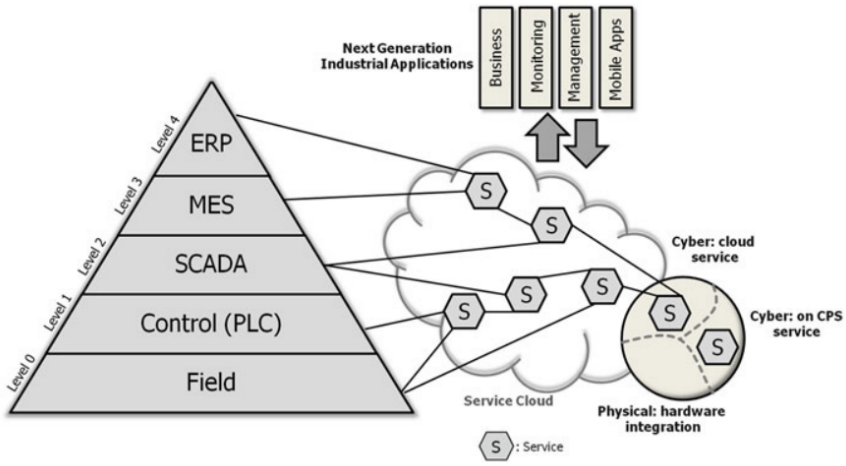

Figure 2.9: Transitioning towards an SOA-based information-driven architecture by offering key functionalities as services (based on [29, 33])

Tackling the "*what to change*" domain, several modern production systems have been addressing transformations through modifications, mainly at the virtual IT level. Many of these theoretical contributions address topics of service reconfiguration in the CPPS scenarios. According to [67], such initiatives intend to push the traditional production systems to be more agile, flexible and reconfigurable, ideally by breaking down the traditional hierarchical ISA-95 automation structure into distributed control architectures, and with non-hierarchical modules linked together through different communication systems. For this reason, the development of manufacturing systems paradigms has evolved hand-in-hand with the exponential growth of technological solutions.

2.2.6 How to Reconfigure

After understanding the reason(s) beyond the modification, there are various possibilities for change that should be made as soon as possible or, concretize them in small adjustments to solve future problems.

Looking at the concepts of the previous subsection (*i.e.*, physical reconfiguration, logical reconfiguration), the most straightforward analogy to do would be the physical reconfiguration through ***manual*** reconfigurations, and logical reconfiguration associated with

automatic changes. However, the idea is to understand (with the help of Table 2.4) that there are other important points to consider and associate.

Table 2.4 describes several points that can be taken into account, such as different ways to reconfigure, with different impacts.

Table 2.4: Physical versus Logical reconfigurations.

Assumptions	**Impact**	**How to reach**	**Reconfiguration Type**	**Trigger**
Plug & Produce	Physical/ Logical reconfiguration	Standard Interfaces (DPWS) Registry and Discovery	Hard/physical reconfiguration	Operator
Dynamic Service Reconfiguration	Logical reconfiguration	Service Registry and Discovery Orchestration and Choreography	Soft/logical reconfiguration	Soft/ Logical, after plugging a component

In order to plug the device into the system, and make it visible in the network, some requirements are needed. Without considering the physical aspects, essential steps [159] can go from the establishment of the physical connection to the discovery of the connected device, until the device configuration is ready. This procedure is performed without the need for manual configurations, via a middleware, *e.g.*, OPC-UA, or by dynamic configuration of realtime network systems [48].

The second reconfiguration case is focused on logical service reconfigurations. Besides the PnP physical devices offered as services, the system processes and tools are also represented as Services. But, in this case, the purpose is to aggregate and compose with other services, aiming to hide the complexity using orchestration capabilities. The idea is to use a discovery mechanism to search for services, compose them according to their objective and the reconfiguration strategy, and then the orchestration process is used to guarantee that all target services are executed in the right sequence. Next it is represented the logical modifications from the orchestrator process level:

- Improvement of the service's behaviour and performance.
- Adapting (remove or add) services from the services' catalogue.
- Changing the structure of a composed service.

In addition of being responsible for the execution of a single or composite service, the orchestrator monitors its life-cycle to check the success of the execution action. The automatic service reconfiguration can leverage from the historical service modifications by improving and replacing the services, either to continue with the same functionality or pursuit a different objective. Collaborative automation depends on raising the highest

possible level of parallel processing. This service reconfiguration will lead to modifications at the planning level, particularly in job reassignment.

In line with this thought [181], one reason for the deterioration of the performance is an inadequate configuration. The proposed methodology to find a configuration that satisfies the desired performance levels is executed based on a centralized scheduler, triggered when the performance indicators are unsatisfactory for the considered timeframe.

The reconfiguration mentioned in IMC-AESOP [84], envisioned an SOA-based SCADA infrastructure to enable the cross-layer service-oriented collaboration among cooperating devices and systems located at different levels. The reconfiguration [84] considers to manually stop the execution and replace the on-line device with a new one. Besides the *post-mortem* reconfiguration strategy, due to the device pluggability, the simulation scenarios to test the configuration expect reactive triggers. The "*How*" challenge is also addressed in SOCRADES [32], which is oriented to the reconfiguration of distributed smart embedded devices. Industrial automation systems allows exploiting SOA benefits at the device and application level. For example the use of DPWS devices worked in collaborative automation [31].

Alternatively, real reconfigurable systems can be achieved through self-organization mechanisms [102]. This approach permits to control the complex system in unpredicted environments, using decentralized adaptations, *e.g.*, service's adjustments and service optimizations [170].

2.2.7 Who Performs the Service Reconfiguration?

The execution of the service reconfiguration process relies on different actors responsible for implementing the necessary reconfigurations actions. As described in Table 2.5 there are two reconfiguration types: manual (*i.e.*, human intervention) and automatic reconfiguration (*i.e.*, software decisions or RMT).

The service composition job can be accomplished in an automatic manner or by human resources using orchestration engines [197]. In fact, both approaches can build a composite service, but the one most common in the practice includes the manual operator [95]. In the automatic physical domain, self-reconfiguration is also possible using automatic Reconfigurable Machine Tools (RMT) or self-optimization tools. Table 2.5 summarizes the types and actors involved in the service reconfiguration.

In opposite, manual reconfiguration forces human operators to perform machine maintenance, to recover it from a failure state and to change tools (*e.g.*, using PnP), in order to perform or optimize the processes.

Table 2.5: Types of service reconfiguration.

Strategy Description		
Who performs		**Reconfiguration examples**
automatic	logical	modifications at the planning or process level, *e.g.*, job reassignment, service optimizations
	physical	modifications using optimizations settings in the current resources performed by automatic Reconfigurable Machine Tools (RMT) or self-optimization tools
manual	logical	manual logical reconfiguration considers human-in-the-loop, by using devices as HMI or other devices (*e.g.*, tablets, smartphones, glasses or gesture devices), assisting the reconfiguration validation
	physical	this reconfiguration considers the ability to manually displace and reassign entities from the production system, *e.g.*, a modification that implies resources changed from their initial physical location

Industry 4.0 enforces human interaction with greater safety and performance, avoiding the location of the human effort in repetitive tasks. Notwithstanding, the human factor has an influence on the reconfigurability capability, particularly in the human-machine interaction, by decision-makers. Given this, the vision is to go from a human-robot collaboration to a Human-CPS integration, *e.g.*, the Operator 4.0 project promises an increase in the quality of work by performing meaningful tasks with automation-aided systems, without compromising the production objectives.

2.2.8 Discussion

Most literature comprehends service reconfiguration, and reconfiguration needs do consider a framework to design and introduce several modifications, mostly reactive using a common monitoring value or pluggability of new devices in the system. This type of thinking is still strongly in-line with predictive maintenance [31, 37, 84, 163].

Nevertheless, the identification of these requirements is crucial for proper service reconfiguration. It does not require to cope with everything mentioned above, but understanding the strengths and weaknesses of the developed solution is essential to predict the feasibility of the upcoming evolution. In practice, performing the service reconfiguration manually is not enough to address the current industrial needs [172], with a genuinely dynamic and pro-active service reconfiguration still being needed.

2.3 Technologies for Dynamic Adaptation Approaches

Not all modifications tasks can be performed by machines. Humans still have the characteristics that computers are hardly ever going to possess, but at the same time, human's intervention tends to take more time and make mistakes. Since it is useless to completely avoid the human errors from the system the idea is to minimize its interaction and to give more attention to the self-adaptive software.

2.3.1 Autonomic Computing

Self-adaptive software (SAS) systems able to adapt their behavior at runtime to keep the desired system performance despite the disturbances of the environment. SAS applications have been identified as promising research that fulfills the necessity of making interventions in systems that require a fast response to adapt while providing a mechanism that lowering the complexity [85]. The literature offers many pieces of evidence [179, 182], of self-adaptive system addressing questions such as *what* to change, *when* to change, which are fundamental to create such a mechanism that reduces human involvement.

For this, a self-adaptive system that is continuously adapting and evolving comprehends many phases, as illustrated in Fig. 2.10.

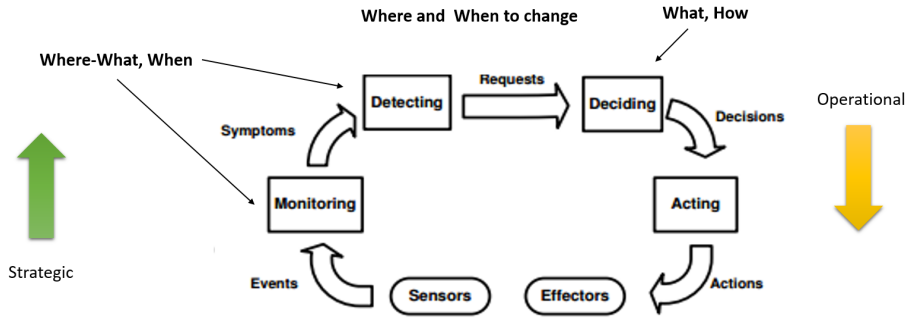

Figure 2.10: Generic Adaptation Loop of a Self-adaptive process with the necessary processes. (*Monitoring*, *Detecting*, *Deciding* and *Acting*.

As illustrated in Fig. 2.10, the adaptation process loop consists basically in four-steps [45, 45, 85] namely:

- *Monitoring* - the environment collects and correlates data from the environment. Conceptually this process is responsible filter the data into behavioral patterns.
- *Detecting* - analyzing the data collected to the internal knowledge. In particular, it associates the *when* with the *where*. In other words, it identifies particular moments to triggering changes in the system.
- *Deciding* the suitable process adapt the system by detailing what resources to change, and how to change it.
- *Acting* the outcome of the decision phase that needs to be implemented, by applying the actions.

These are essential steps of any Self-adaptive System. By designing these processes in the form of continuous loops, allow taking the system's output/actions into consideration to adjust the future actions towards the target value at runtime. These technological advances in fields of Self-adaptive Systems (SAS) that allow to minimize the possibility of human errors and also promote mechanisms for controlling, adapt, and evolution of software raise some attention to well-known companies like IBM, which create the manifesto coined "Autonomic Computing" (AC) [3]. The manifesto illustrates an analogy

between the concepts of “reasoning” that is divided between “reflex” and "thinking" in the autonomy of the nervous system of the human body and an autonomous software system. Based on the success of this document, in 2003 the vision of AC was elaborated by Kephart and Chess. The authors describe four general self-management properties [85], which a system must own under the AC vision.

2.3.1.1 The Self-* Properties

The autonomic computing vision that defined by IBM consider four general adaptative properties that a system should have to be considered self-managing, known as the self-* (*i.e.*, self-management properties) that include *self-configuring* (capability of automatic configuration of components), *self-healing* (capability of automatic discovery, and correction of faults), *self-optimizing* (capability of automatic monitoring and control of resources to ensure the optimal functioning) and *self-protecting* (capability of proactive identification and protection from arbitrary attack) [85, 182].

Despite these four fundamentals self-* properties from the original initiative, an extended view was proposed to add more capabilities to the general autonomic computing paradigm, *e.g.*, Self-(adjusting, aware, diagnosis, protecting, simulation) among many more) [94]. In fact, many of the self-* properties can be connected to sub-properties. For example, the implementation of the *self-healing* may include the process of being capable of analysing itself to identify existing to anticipate potential issues (*i.e.*, *self-diagnosis*). (for further details, see [2, 94, 138]). All these characteristics affect the degrees of autonomy of the system, the features chosen to increase or decrease the system’s ability to adapt and become more or less flexible. In fact, some authors [79] argue that these self-* properties of autonomic systems are inspired by the properties of software agents, which Wooldridge and Jennings defined in 1995 [219].

2.3.1.2 The Autonomic Control Loop

One of the breakthroughs in the design of AC presented by IBM‘s is MAPE-K (see Fig. 2.11). This is the de facto paradigm of the architecture to design self-adaptive software [85]. The development considers a notion of autonomic element architecture (see Fig. 2.11).

MAPE-K is a common paradigm among the Software engineering community to develop self-adaptive systems because of its simplicity based on (monitor-analyze-plan-execute over a knowledge base). These processes permit to control the system in an autonomic manner, acting more reactive or more reflexive mode. Although a notion of autonomy is strongly connected to the MAPE-K model, not all autonomous systems need to be directly designed with this logical architecture. The model describes how to organize concepts, not how they should be drawn. In this sense, any software engineering should identify the concepts and consider them in the construction phase of an autonomous manager.

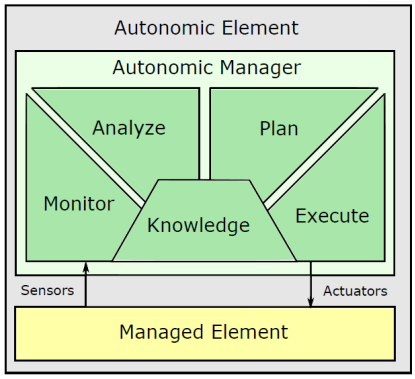

Figure 2.11: Autonomic computing MAPE-K references model (Source: based on [85]).

2.3.1.3 Artificial Intelligence

Artificial Intelligence (AI) and Software Engineering (SE) are two subfields of Computer Science (CS), that share the same goal of solving problems automatically. In fact, some authors point out the relationship between AI and software engineering, particularly in the development of autonomic computing systems [76]. One objective in this work is to empower software systems in the way that support the engineering a self-adaptive software to function effectively. Intelligent Software Agents has the potential to make a remarkable contribution to the realization of this self-adaptive software.

2.3.2 Agent Technology, Agents and Multi-Agent Systems

The concept of an agent is neither unique nor consensual. From the Computer Science perspective, is a computer program that use AI technology to automate specific tasks, but many interpretations can be found in the literature due to the broad nature of the agent concept. Some proposed definitions found in the literature are summarized below:

- "an agent is a computer system that is situated in an environment, and that is capable of autonomous action in this environment in order to meet its design objectives" [217].
- "an agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors" [180].
  "an agent is a computational entity that can be viewed as perceiving and acting upon its environment, that is autonomous and that operates flexibly and rationally in a variety of environmental circumstance" [212].
- "an agent is a persistent computation that can perceive its environment and reason and act both alone and with other agents. The key concepts in this definition are interoperability and autonomy" [190].

The broad nature of the agent concept is not only the responsibility of the many approaches focusing on the distinct attributes. Over the past decades and more, the discussion of this subject in academia contribute to a natural evolution of concepts and paradigms in Agent Technology. One of the promising paradigms is the multi-agent system paradigm that comes out from the Distributed AI (DAI) field [59, 217], also called collective AI. Typically,

the multi-agent system involves a set of agents interacting to fulfill one or more tasks. Each agent considers different properties, such as intelligence, autonomy, pro-activeness, adaptation and social behaviour [217], these differences, in terms of properties adopted, create a distinct type of agents. In general, all of them are important, but depending on the objectives, their importance can increase or decrease.

According to several contributions [4, 59, 217] the properties of a agents are:

- *Scope of action*: the scope of action limits the application of the capabilities of an agent. Accordingly, the degree of flexibility of a technical agent is determined by a given scope of action.
- *Autonomy*: attribute of an agent, which allows it to control its internal state and its behavior. Through autonomy, an agent acts and makes decisions based on its local knowledge and activities.
- *Encapsulation*: attribute of an agent which allows it to control the access to its individual constituents which are not visible externally. State, behaviour, strategies, and objective represent the individual constituents that are encapsulated within an agent.
- *Stateless:* the services cannot preserve any new information from past experiences.
- *Goal-orientation*: attribute of an agent which allows it to orientate its behaviour to one or more objectives, which it attempts to accomplish.
- *Reactivity*: the capability to sense the environment and to react accordingly.
- *Persistence*: the capability of an agent to keep its internal state during its lifecycle.
- *Interaction*: the capability of an agent to interact with other agents, in order to accomplish individual objectives or to manage dependencies among each other. The basis for the interactions among agents is a shared semantic, an underlying organizational context and a common terminology model, which are together referred to as an ontology.

In agents, autonomy and cooperation have a particular prominence: the autonomy can represent the ability to perform decisions without human intervention, specifying different levels of autonomy; cooperation is the capability to interact with each other, acting together to achieve a global system goal or shared goals. Thus, modelling cooperation arises naturally, reflecting the need for accomplishing a global task. Unfortunately, the literature is full of examples of bad agent-based solutions, which are implemented as computer programs, having logical layers with no autonomy or intelligence [69].

A multi-agent system is a society of agents that represent physical and logical objects of a system. The global system behaviour can be achieved through the collaboration and interaction among the individual agents, each one having its objectives and behaviours, and possessing its perceptions and cognitive competencies.

In multi-agent systems, each agent has only a partial view of the system. The development of an agent-based control system usually follows a bottom-up perspective. This

paradigm introduces several advantages when compared with the traditional approaches, *e.g.*, more robust solutions are reached since the control functions are distributed over the agents' network, thus some of the critical failures, like bottleneck problems, which are mostly associated to centralized systems, no longer exist.

2.3.2.1 Agent Communication

In order to achieve a common goal since several agents distribute the knowledge, the agents need to communicate. The exchange of messages supports such interaction that is founded using Agent Communication Language (ACL), which pretends to transform the messages exchanged more interoperable.

The two major agent communication languages are KQML (Knowledge Query and Manipulation Language) and FIPA-ACL [65]. These specifications, KQML and FIPA, are very similar, they both deal with the messages in order to achieve a mutual understanding of exchanged messages using speech act theory [187]. The KQML is the first best-known language since it is the first that emerged, compared with FIPA. At its foundation, it supports two different contexts, a formalization of the message format and a message-handling protocol [59, 64].

2.3.2.2 Learning

Learning is an essential ability of every intelligent being. Learning allows acquiring knowledge for predicting and planning further actions based on observations of the environment, analogies, examples or experience. Learning makes an entity more dynamic to evolve and to improve the system's ability to act in the future. Among several learning definitions, was chosen which is more interesting in our work. Mitchell, presents "Learning is improving automatically with experience" [134].

In literature, various examples propose several classes on learning in agent domain, for example, agents guide by the goals or agents guides by the action that maximizes the expected utility of the action, among others. For example, if an agent creates a new service with a high level of trust, or even if it is necessary to manage our agent network, it can be followed an incentive mechanism to accomplish a better trust flow. A possible approach is to create an architecture of incentive mechanism that could be RL and using, for example, the Q-learning algorithm.

When designing agent systems, it is impossible to predict all the potential conditions and situations that an agent may encounter. Therefore, agents have to learn from past iterations and adapt their own skills and behaviours, to be able to act with the unpredictable environment characteristics. RL allows us to create a predictive system [26], which could help us to make better decisions.

2.3.2.3 Ontologies

Distributed and autonomous agents only have a partial view of the entire system, requiring the need to interact among them to exchange shared knowledge. Also, since the entities will work under a very dynamic and open environment, they must understand each other. For instance, similar services can represent different scenarios. In this way, the use of ontologies is crucial to guarantee a common structure of the knowledge exchanged among the distributed agents during their conversations. Ontologies can increase how knowledge is expressed. Moreover, ontologies can increase the semantics of the services, like the processes, how the knowledge is exchanged between two entities where both can understand the meaning [141, 164]. The term ontology defines the vocabulary and the semantics that are used in the communication between distributed entities, and the knowledge relating to these terms.

It is mandatory to use a meta-model to exchange interoperability, and these Meta-Models can be made in several formats, namely in XML [222], RDF or OWL (Web Ontology Language) [130]. With ontologies, it is possible to share properly structured information without compromising the semantic issues that may exist. The sharing of information based on ontologies in a distributed system produces an environment very easily malleable and with highly agile customization. A good base of knowledge representation of the case study allows supporting better the decision-making processes, having a large impact because the system can adapt or fix errors without human interaction.

2.3.3 Cyber-Physical Systems

2.3.3.1 Key Concept

The term Cyber-Physical Systems (CPS) promote the coexistence (*i.e.*, the integration and coordination) of Cyber and Physical counterparts in a network structure to perform the system's functions collaboratively. This term, originally coined by US National Science Foundation [139] describes systems that consist of the integration of the Cyber "software" element with the Physical elements "means interacting with the physical world". This concept represents a promising system to bring many opportunities, like the possibility to adapt, collaborate, and evolve altogether, or partially the cyber or the physical world element in a distributed form.

The communication takes an important part since the elements of the CPS are usually distributed and cooperate to produce some global behaviour. Similar to the emergent behaviour Agent Technology that arises from the collaboration among many agents the combination of several CPS rises evolving properties, like in Agent Technology improvement in the coordination is necessary to ensure that there is no undesirable or unexpected emergency. More recently, with the appearance with Industry 4.0, this term combined with the possibility of human-machine interaction [195], this idea was explored as the "Operator 4.0" by [177].

2.3.3.2 CPS - Opportunities and Challenges

At present, there is significant attention in the CPS, for example, the millions of euros invested in EU research projects [162] which represent the confidence of the benefits of CPS to companies. In fact, in the development of the next generation of industrial systems, CPS is a key pillar in the Factories of the Future public-private partnership financing in €1.15 billion, with the overall purpose of developing and integrating the leading support technologies supporting innovative and adaptable machines and systems. Studies have shown that CPS technology can generate new information to help companies improve processes and make decisions faster and with a lower error raterate [67, 98, 99].

As expected, the integration of CPS in industrial systems proposes great perspectives towards agility and system sustainability, and reconfigurability [98]. Identical to Agent Technology, CPSs are distributed by nature, also influencing domains in common for a variety of purposes (*e.g.*, traffic management, tourism, manufacturing, and health). With this in mind, the European Roadmap on Research and Innovation in Engineering and Management of Cyber-Physical Systems [6] has distinguished the CPSs features into:

- Large, often spatially distributed physical systems with complex dynamics;
- Distributed control, supervision, and management;
- Partial autonomy of the subsystems;
- Dynamic reconfiguration of the overall system on different time-scales;
- The continuous evolution of the overall system during its operation;
- Possibility of emerging behaviours.

Further related to this, Santos *et al.* [183] have raised an interesting question about EFFRA reports [55]. Even though EFFRA announced back in 2016 that the strategic discussions for the next work programme have expecting to that will be more related to Industry 4.0, in fact, the report was named "Factories 4.0 and Beyond", in which the vision laid out in the Industry 4.0 and CPS.

Some of the mentioned features are significant in the CPS applied to manufacture. Important advances of CPS concluded by J. Lee *et al.* [98, 99] point out that more research is indispensable, the author proposed 5-level CPS structure, namely the 5C architecture, this initiative provides a step-by-step guide for developing and deploying a CPS for manufacturing application. In this CPS architecture, the authors suggest a cyber-manufacturing system that involves different technologies.

2.3.3.3 CPS Industrial Standards

As to the implementation of CPS, since CPS solutions involve different hardware and software technologies, is crucial the standardization in the development of CPS to accomplish the industrial requirements and enabling a smooth migration. Given this concern, a dedicated IEEE Working Group was created named "IA Software Agents - Software

Agents Low-Level Automation Functions", with the idea to provide a set of best practices for the integration process, aiming to standardize the interface process to allow the reuse and transparency [1]. Therefore, seeking wider acceptance of the CPS by industry, more research is necessary on the standardization of CPS.

2.3.3.4 Developing CPS for Industrial Systems

From the industrial perspective, the CPS are modern production systems assisting in the design of complex systems, which consists of many distributed elements like Cyber-Physical Components, that can process the production data and make some decisions autonomously for these CPS consider intelligence, responsiveness, and adaptation, like Smart Services and Smart Manufacturing. This idea is nowadays making inroads in the engineering of systems of production presupposing systems of cybernetic physical production.

2.3.4 Service-Oriented Architecture

Service-Oriented Computing (SOC) and SOA have been subject of studies for quite some time now, offering an excellent solution to the many challenges of the current business, namely providing agility through quick response to changes, able to easily and dynamically adjust business rule, allowing companies to save time and money. One of the definitions that seem to fit a greater number of researcher states that [51]:

"*SOA establishes an architectural model that aims to enhance the efficiency, interoperability, agility, and productivity of an enterprise by positioning services as the primary means through which solution logic is represented in support of the realization of strategic goals associated with service-oriented computing*". Thanks to the efforts in these last years by the SOC community, several aspects in this definition provide paths to facilitate the emergence of new services by minimizing the relation of dependencies, in a flexible and dynamic approach reusing existing services and avoiding then the creation of redundant services [147].

2.3.4.1 Key Principles

There are key principles of SOA to creating an effective service-oriented solution, as enunciated in [51] these are:

- *Abstraction*: the logic and its complexity are compressed and offered as services, being its complexity hidden.
- *Reusability*: services can be easily reused later since the complexity of the services is divided into several services.
- *Composition*: services can be easily reused later since the complexity of the services is divided into several services.
- *Stateless*: the services cannot preserve any new information from past experiences.
- *Service contract*: represents the communication protocol agreed.

- *Loose coupling*: the relationship among other services, depending on the business logic created at design time, a set of services with minimal dependencies is desirable. This principle means that services are designed with no affinity to any particular service consumer.
- *Discoverability*: the services have descriptive information that allows them to be found and accessed through a discovery mechanism.
- *Autonomy*: represents the business logic associated with the functionality of the service.

A common mistake is to think that the SOA paradigm is only based on the 3-step processes, discovery, request, and response, as it is described in Fig. 2.12. This is not completely true, but most of the benefits arise from using such an approach where the three major entities are illustrated as well as their own actions. For example, an elementary SOA application involves two entities: a service provider and a service consumer. The provider exposes and supports the service while the consumer invokes the services to pursue its own goals.

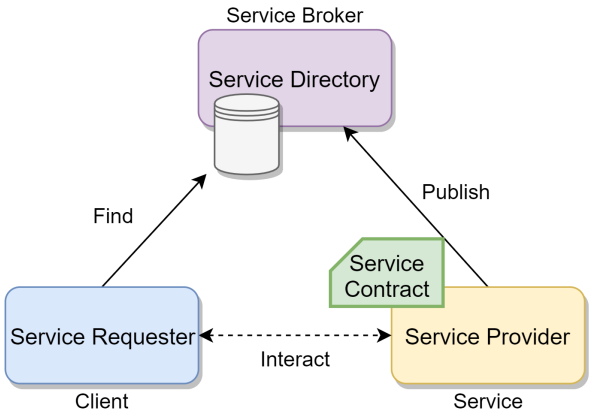

Figure 2.12: Service-Oriented Architecture.

Figure 2.12, exemplifies the parties and communication phases as follows:

- The first action is the registration process (*step 1*), where the provider registers the services that it can offer.
- A discovery process (*step 2*) of finding the services that provide the required functionality is usually called UDDI (Universal Description, Discovery, and Integration).
- The service consumer receives from the UDDI Registry the interface needed (*step 3*) to invoke the service provider (*step 4*).

SOA principles are well integrated because there is good incorporation with collaborative automation, in the sense of self-governing, reusable and loosely coupled distributed components. Also, due to this effort, modelling tools with Web Services protocols were developed to deal in a more abstract way, from the BPEL (Business Process Execution Language) [36] to the WS-BPEL (Web Services Business Process Execution Language) [87], and WSFL (Web Services Flow Language) [116] proposed by IBM. The purpose is to assist

in the modelling, which should be so abstract that if it inserted a new device and new processes in the automation system, they could integrate in order to achieve the objective: total integration. Some process can become automated to assist in the modelling, such as:

- *Orchestration*, which is an automatic and coordinated management of services, taking into account a set of centralized services into a single one. In other words, it consists of a combination of services to produce more complex and useful services.
- *Choreography*, which describes each service as a service that knows exactly when to become active and with whom to interoperate, in a collaborative way.

Both specifications should be implemented to make a more autonomous system.

2.3.4.2 Middleware

The middleware plays a key role in connecting several pieces of complex distributed software together in one common platform in order to achieve multiple goals. Researching this topic revealed that most middleware functionalities focus on providing vertical and horizontal integration connectivity with other architectural components in a way to integrate all the different systems and make them work together in a homogeneous environment. However, other reasons reveal interesting to use middleware in the following situations:

- *Situation 1:* establish a link between a client and a server application for data exchange.
- *Situation 2:* provide discovery mechanisms to discover features provided in the system dynamically.
- *Situation 3:* provide data translation functionalities from one data format into another.

Even though the general purpose of middleware is to facilitate the communication between applications (*i.e.*, Situation 1-2), in some instances can a single application to connect to systems (*i.e.*, Situation 3), in fact, this situation the middleware can be considered as an adapter [62]. Finally, and after the initial evaluation of some principles and technologies, the aim now is to focus on existing work.

2.4 On the Road to the Factories of The Future

In this section the ongoing research on intelligent reconfiguration in the Smart Manufacturing and Industry 4.0 initiative will be reviewed, with the identification of technology trends and key design principles, offering a strategic roadmap to understand the current position of these topics in the Industry 4.0 phenomenon.

Over the last years the fourth industrial revolution has been gaining attention due in part to its obvious benefits, and how easy it is to perceive the importance of emergent technologies and paradigms (such as, distributed artificial intelligence and SOA) to assist in this initiative [102].

Although the priority of this thesis is to explore adaptive and intelligent manufacturing topics, it is worth mentioning that in the factories of the future report [55], it is often pointed out along the report that there are several megatrends that are not directly related to smart manufacturing, such as, social sustainability or knowledge workers. Even focusing only on some topics of interest of this thesis in Industry 4.0, it is possible to highlight the importance of Industry 4.0 over the years in the academic world through an analysis of the evolution of the number of scientific publications associated with Industry 4.0.

2.4.1 Research Methodology

This section overviews the existing scientific contributions on intelligent manufacturing topics, aiming to identify the scientific publications that recognize the concepts discussed in chapters 2.1 and 2.2. The methodology developed for this analysis was inspired in [61, 203].

From a general perspective, the dataset extracted was used to analyse the current status of smart manufacturing, as an attempt to predict its evolution within the Future of Industry 4.0 and the Fourth Industrial Revolution.

The first step was to restrict the literary research to smart manufacturing topics only. After defining the scope, the research was built to include published papers in leading international journals and indexed in recognized technology databases, namely Science Direct and Scopus repository. The search included the following keywords: "*industry 4.0*" OR "*smart manufacturing*" OR "*fourth industrial revolution*", as part of the abstract, title, or keywords of the manuscripts. Additionally, only English-language manuscripts published from January 2010 to August of 2019 were considered. As a result, a collection of 6871 relevant documents was obtained, which are represented in Fig. 2.13.

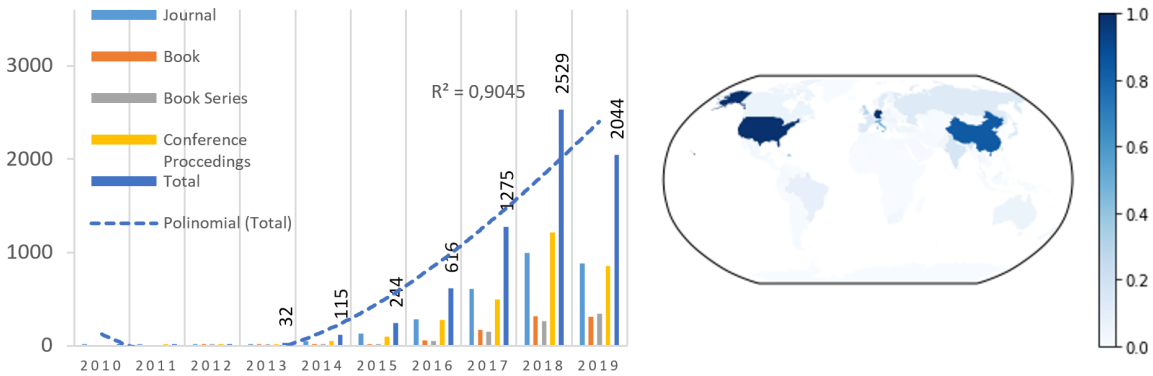

Figure 2.13: Number of publications related to Industry 4.0, their type and geographic distribution.

The performed analysis allowed to witness a growth trajectory on the number of publications discussing Industry 4.0 and the Fourth Industrial Revolution (4IR) throughout the time considered (Fig. 2.13, left-side). This suggests a growing interest in these topics along the last years and, considering the generated trend curve, the scientific production on this

topic will most likely continue to evolve in the near future.

When analysing the affiliations of the publications' authors by their Country/Region it was also possible to observe that the Fourth Revolution is having a heterogeneous impact across the globe (Fig. 2.13, right-side), being predominant in the USA and Germany and followed by China. Notwithstanding, it is easy to predict that the digital revolution in manufacturing (known as Industry 4.0) will evolve globally given the increasing financial investment on its technologies. In Europe, for example, companies have been receiving huge amounts of capital to make this industrial vision feasible [25]. EFFRA reports that the overall resulting size of the Factories of the Future programme within Horizon 2020 will reach €7 billions [55], estimating that €500 million/year will come from the public funding budget. This demonstrates that local governments are taking in high consideration the need for initiatives, strategies and research programmes that would enable this digital transformation.

Similar Initiatives

Although the "Industry 4.0" initiative was born in Germany, its economic and social impact have been encouraging a worldwide development. Examples include the "I40" in Portugal, "*Industria Conectada 4.0*" in Spain, "*Industrie 4.0*" in Germany, "*Piano Industria 4.0*" in Italy and "*Smart Industry*" in the Netherlands. A similar development also occurs outside Europe, where the R&D investments are also strongly supported, for instance USA with "*Industrial Internet*" with impact into Smart Manufacturing and "*Made in China 2025*" in China.

However, Industry 4.0 is not just a matter of government initiatives, around the world companies, such as Accenture, McKinsey, and PwC are also actively involved. PwC analysed the opportunities and challenges of Industry 4.0 [88], and estimates that the digitized products and services generate approximately €110 billion of additional revenues per year for the European Industry, an economy that, in 2009, around one in ten (9.8%) of all EU-27's non-financial business economy, representing almost 2 million enterprises classified as manufacturing [55].

2.4.2 Analysis of the Results

For a deeper analysis of the most important publications, based on the number of citations, an author's co-citation map was built. The co-citation occurs when two papers are cited by a third paper [191], based on this, the co-citation analysis helps to understand who are the authors more interested in a specific topic, how these authors cooperate and also identify contributions likely to be closely related [145]. To do so, it was created a co-citation network map to easily understand this analysis as illustrated in Fig. 2.14.

The network map shown in Fig. 2.14, the references to the authors were considered through the representation of several nodes. These are, in size, directly proportional to

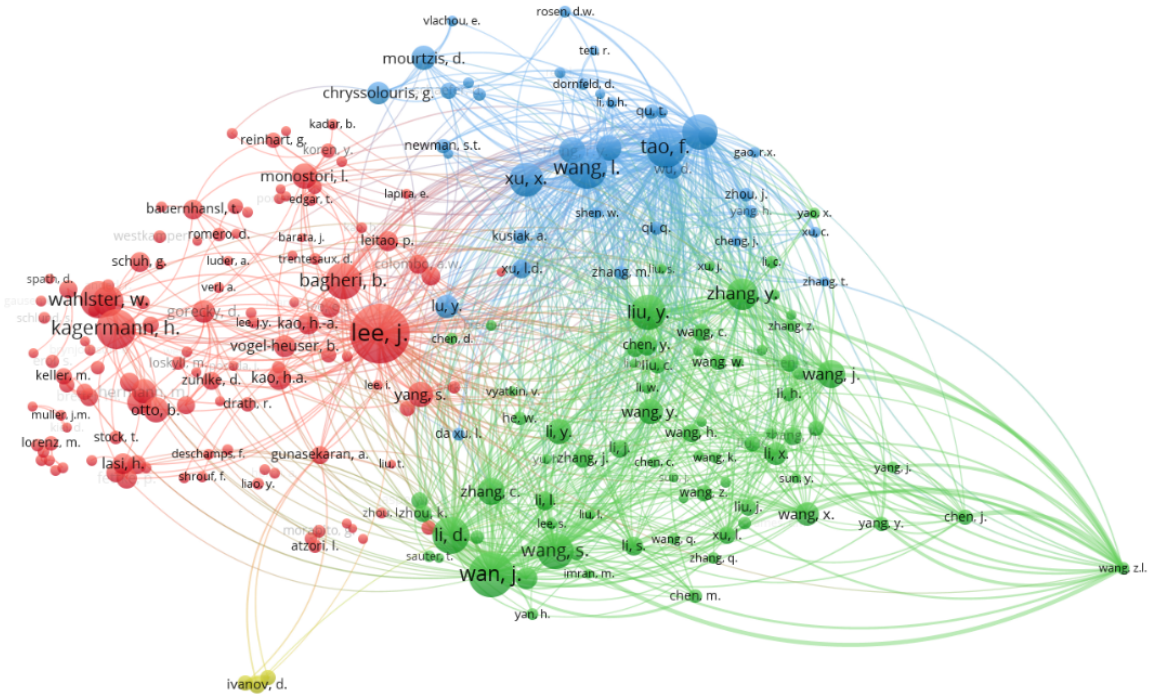

Figure 2.14: Author co-citation network map of the literature on smart manufacturing (time-frame from 2010 to 2019; n = 162377 authors in the co-citation network; threshold 123 citations per author, display 239 authors). The co-citation link is based on the "cited authors" item.

the number of citations corresponding to each of the authors concerned [200]. It was also considered "*full counting*" as the counting method and displaying nodes that have at least 123 citations per author. Looking at the co-citation map pictured in Fig. 2.14, we can see that our dataset on smart manufacturing has a 4-cluster network.

The first cluster, illustrated by the colour red, presents as most cited authors J. Lee, with relevant work on CPS, and Kagermann, who is a reference in the Industrie 4.0 initiative, clearly reflecting the importance of their work in the rest of the community. Other relevant scientific contributions in the area, in particular SOA, are also considered to be in this network by authors such as A. Colombo and K. Stamatis. The remaining clusters, represented by other colours, constitute communities that refer to different topics, less relevant under the scope of this thesis. Despite the co-citation's dispersion observed, it is easy to identify the predominant authors, as co-citations cores (such as Lee. J, Kagermann, Ivanov. D., Tao F.). However, the dynamics of the graph depicts different similarities, reflecting the existence of distinct scientific production lines within each cluster obtained from the Science Direct and Scopus dataset.

2.4.3 Industry 4.0 Design Principles

While the Industrie 4.0's vision is one of the most discussed topics these days [77], it is evident that companies are still struggling to address the opportunities offered by Industry 4.0. When it comes to identifying and formulating key strategies to implementing the

Industrie 4.0 principles, several articles explicitly address design principles on this topic [77] to understand how to get the maximum potential from the Industry 4.0 vision. This allows for many authors in the communities (see the group in red colour in Fig. 2.14), to share their view of design principles and strategies for the implementation of Industry 4.0 systems, addressing various challenges such as SOA, MAS, and CPS.

In a quick look at the co-citation network map of Fig. 2.14, it is understood that the size of the nodes indicates their impact on the network. Within the spectrum of the most significant nodes, we can identify suggestions for design principles [77, 83] on how to adopt a smarter type of manufacturing. Usually, the mentioned principles comprehend six main design principles, which appear as the most common principles (see Table 2.6).

Table 2.6: Most common principles (Source: based on [77]).

Design Principle	Brief Description
virtualization	enables a virtual copy of the physical world can be created, where the physical processes are linked to virtual plant models and simulation models
real-time capability	real-time capability supports reacting fast decisions if the necessary that data is collected and analysed in real-time.
Interoperability	Connectivity between people, machines, and systems are crucial to facilitate collaboration under specific standards with each other.
service orientation	permits the manufacturing services to be utilized and integrated across (verticaly and horizontaly) manufacturing platform.
decentralization	to enable the break from the centralized control into an Industry 4.0 decentralized modern approach, where machines can locally and autonomously make their own decisions
modularity	facilitated adapt to changing requirements by replacing or expanding individual modules

When bringing together the application of these principles in a methodology, it is easy to create and connect a network of machines and humans, where a modular structure can monitor the physical processes, creating a virtual copy of the physical world and make a decentralized decision promptly, increasing the system's efficiency.

Reference Architectures for Industry 4.0

When technological trends begin to be understood on a large scale due to their apparent benefits, it is natural that there is a search for standardization and adoptions of good practice, which is referred to as a *Reference Architecture*.

The main idea behind the Reference Architecture is to structure enabling technologies and put them in the context of Industry 4.0 for practical use, *i.e.*, serve as a mechanism for capturing and transferring knowledge. To make sure that all entities involved in the fourth industrial revolution have a common framework to understand each other's activities, the two leading organizations leading this transition process of the industrial revolution (*i.e.*, Industrial Internet Consortium (IIC) and Plattform Industrie 4.0) have proposed two

reference architecture models, as follows:

- **The Industrial Internet Reference Architecture Model (IIRA)** focuses on the Industrial Internet of Things across industries, highlighting cross-industry commonality and interoperability.
- **The Reference Architecture Model for Industrie 4.0 (RAMI 4.0)** is related to manufacturing and related value-chain lifecycles.

German Electrical and Electronic Manufacturers' Association (ZVEI) proposed the RAMI 4.0 reference architecture model (see Fig. 2.15) with the main objective to support Industry 4.0 initiatives.

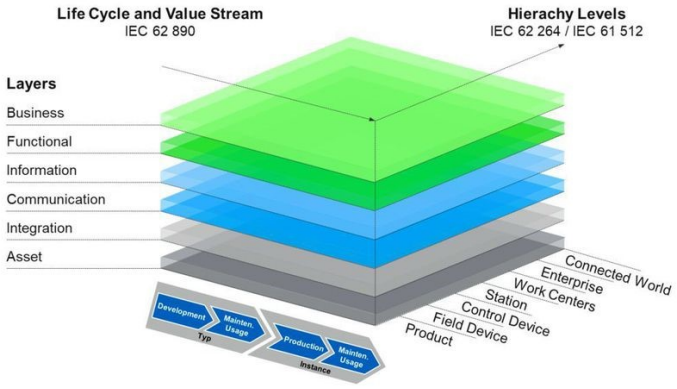

Figure 2.15: Reference Architecture Model for Industrie 4.0 (RAMI4.0) (Source: from [201]).

The RAMI 4.0 offers a format that adheres to the exchange of information throughout the production chain addressing specific standards. This architecture model is represented in a cubic layer model (see Fig. 2.15), that compares the life cycles of products, machinery, factories, or orders with the hierarchy levels of I4.0.

However, it turns out that just understanding the model and proving its technical advantages are not the hardest parts. Empowering companies in all aspects to successfully implement these concepts is the real challenge that requires a mindset shift in the existing organizational culture. Although it is a bit premature as it is relatively recent, companies must be well placed to stay focused and be aware of this kind of references that enable and support them in introducing and integrating solutions. They must also be aware of application standards, scenarios, and solutions that must be taken into consideration in the future.

After finding the most commonly used principles in the literature on how to create a full or partial transition to Industry 4.0, the research is conducted into the technologies needed to create a solid foundation for the flexibility and reconfiguration requirements.

2.4.4 Technologies Principles

Besides designing methodologies to enable and improve intelligent manufacturing performance, the technologies necessary to support and drive these methodologies must also be defined. How often a keyword is used suggests information about the type of technologies authors use in their research. Considering this, Fig. 2.16 summarizes the frequency of some technologies and paradigms used for Industry 4.0 in the collected dataset.

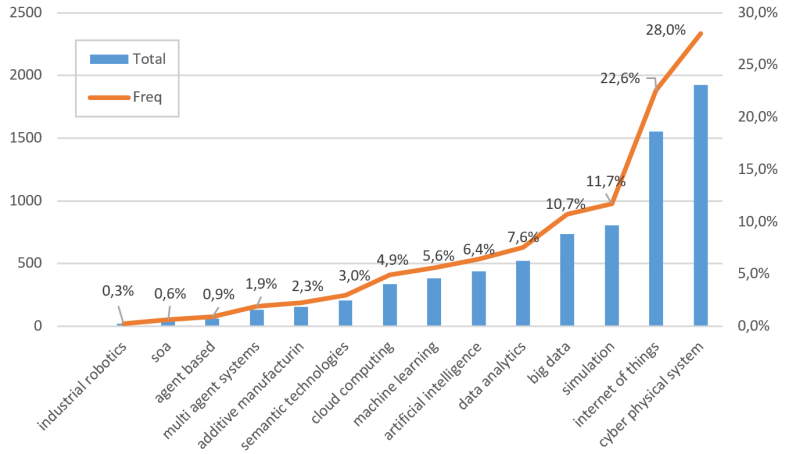

Figure 2.16: Analysis of the frequency keywords used in the Industry 4.0 domain.

The analysis of the most frequent keywords allows concluding that CPS is the most significant one, unlike other technologies that have been around for a long time, such as SOA or MAS. Instead of searching for the frequency of keywords that were already defined, the goal now is to understand which paradigms and technologies are most stated in the dataset, without referring them *a priori*. Knowing that the most common topics and technologies approached in the analysed articles can be observed from keywords, we thought it would also be interesting to understand how they could correlate. For that, the VOSviewer software[2] was used for the generation of a co-occurrence map (Fig. 2.17), which allows visualizing the similarities in the dataset and the dynamics between keywords.

The most frequent keywords (that potentially refer to the thematic of the articles) can be observed through cluster analysis in Fig. 2.17. From the 4706 keywords, only 28 keywords are selected and displayed in Fig. 2.17, which corresponds to the keywords with a minimum of 20 occurrences per keyword. Despite showing the 28 keywords it is possible to see how they are divided into groups and connected. In addition, as can be noticed, the strong bond between two nodes means that there is a more significant co-occurrence between them. This type of connection can be visualized among the keywords, which means that they are used together more often, as an example, "*cyber-physical systems*" and "*internet of things*" are two keywords that appear to have a strong connection with the keyword "*industry 4.0*" in relation to the others.

[2] www.vosviewer.com

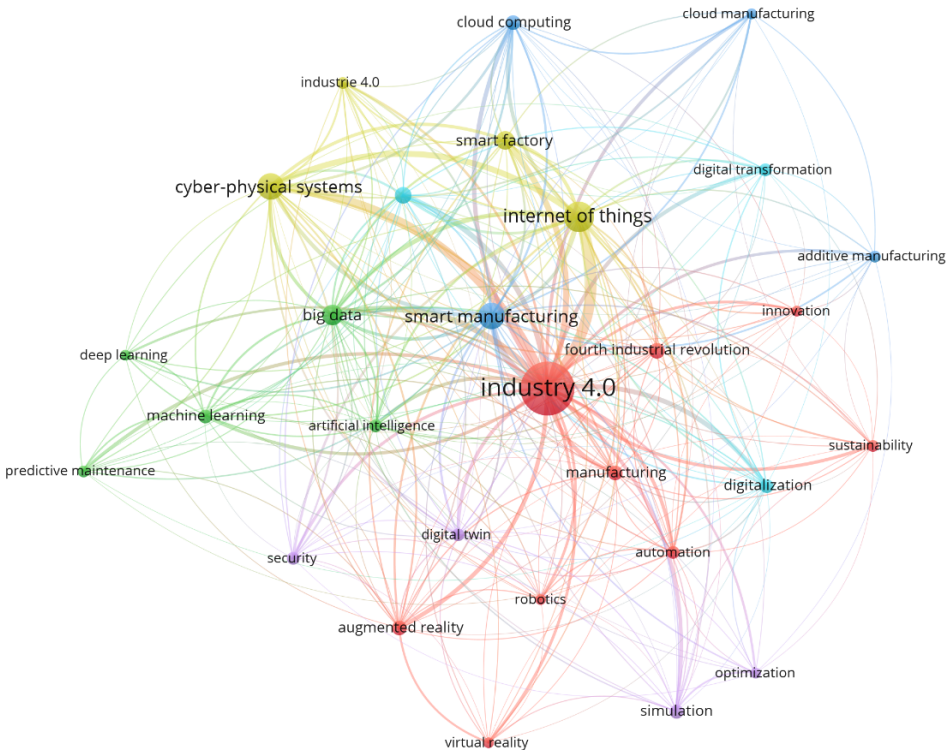

Figure 2.17: Co-occurrence clustering map of author keywords (2010-2019; n = 4706 keywords; threshold 20 occurrences per keyword, display 28 keywords). The size of the nodes indicates the frequency of the keywords. Links between two nodes indicate the strength of co-occurrence between the keywords.

Therefore, design methodologies, including the corresponding tool support, must be defined to facilitate the modular development of system-level services implementing causal chains of physical, technical, and organizational processes.

Next, using the same dataset, we review the ongoing research on the Industry 4.0 phenomenon, highlighting its key design principles and technology trends, identifying its architectural design, and offering a strategic roadmap that can serve manufacturers as a simple guide for the process of Industry 4.0's transition.

2.5 Requirements & Principles

So far, significant questions have been raised to design a dynamic and reconfigurable system [160]. In this stage, several conditions are gathered in order to consolidate the requirements necessary for dynamic reconfiguration of services in intelligent manufacturing systems.

In Fig. 2.18 are illustrated some actors and functionalities for better illustrate the requirements that aim to push forward to a generic reconfigurable and flexible system into an intelligent reconfigurable manufacturing control system. Mainly for the design of a service-oriented multi-agent system to instantiate the CPS.

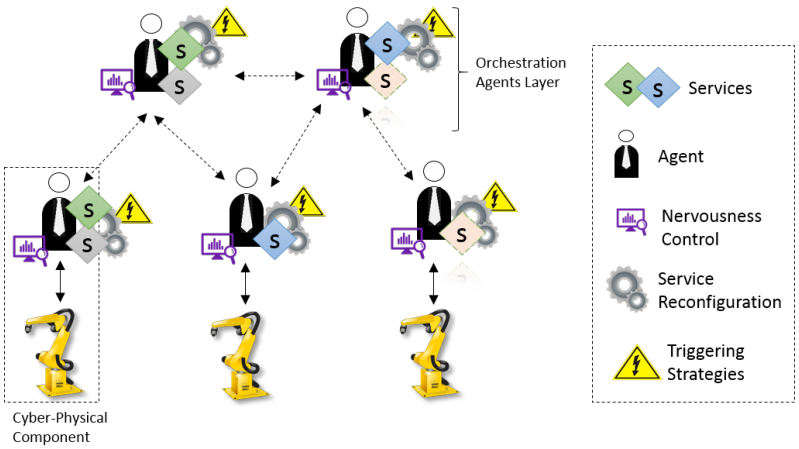

Figure 2.18: Actors and functions for the service reconfiguration.

Aiming to execute a truly dynamic, intelligent and pro-active service reconfiguration, considering the referred reconfiguration types, the specific solutions based on technological (or non-technological) requirements need to be observed:

> **Requirement 1 (R1): The opportunity to execute a service reconfiguration must be identified internally (regarding the system), automatically and at runtime [175].**

This requirement is related to the internal behavior that each distributed entity should have to detect reconfiguration opportunities. The continuous monitoring of the industrial resources, during its execution process, promote several predictable cases but unforeseen events during the time. For guarantee this requirement, the following principles are designed considering the "*When*" and "*Why*" questions:

Principle: Autonomous Triggering Strategies each distributed entity is individual and autonomous to enforce service reconfiguration. The ultimate decision in acting over the physical automation or orchestrate a set of services and/or other entities is identified internally and automatically according to different triggering.

These strategies are crucial to support a truly and efficient reconfiguration process based on the analysis of the available set of services performed and the hardware environment observations, *e.g.*, these strategies can be imposed in an autonomous and proactive multi-agent system solution. In [169] it is suggested a model that analyses three types of strategies to detect possible situations to reconfigure, namely: event (*e.g.*, failures), periodic (*e.g.*, verify the current service performance) and trend (*e.g.*, tendency or pattern in the degradation of service performance). Thus, for a quick identification truly dynamic reconfiguration strategies must be performed automatically, without human interference and after the system start-up (online), during its execution (runtime).

> **Requirement 2 (R2): The system needs to have the capability to select an alternative reconfiguration solution and perform the reconfiguration on-the-fly, reducing the impact of condition changes (*e.g.*, a performance degradation) [175].**

The problems associated with bad decisions in systems that give this power to the operators, in particular to unqualified operators, should be avoided. The best methodology in these situations is to support the operator with pieces of advice and information rather than to additional and unnecessary judgment capability. In this sense, and according to this requirement, the reconfiguration mechanism is designed in a way to select reconfiguration alternatives on-the-fly automatically. This can be reflected on a mechanism that explores alternatives just like the new reconfiguration opportunities at runtime, and then decide on the best solution to keep the system efficient. To do so, the following principles assure to empowering this requirement:

Principle: Evolution implies a gradual process of improvement in runtime, this means, refining the process by changing any of the available entities using the dynamic reconfiguration actions. The proper evolution process, subject to reconfiguration actions determines what the service reconfigurations to implement in a smooth manner are. This implies the elaboration of a pool of possible alternatives for the service reconfiguration. One of the ideas to reduce the impact is to make small modifications over time (self-optimization). This type of approach proposes a continuous and incremental model to improve the process on-the-fly considering the "when" and "how" questions.

Principle: Global reconfiguration welfare the opportunities to reconfigure will be subject to collaboration automation principles the distributed entities aim to maximize the local gain but also important is to assure or improve the global system performance based on the “who”.

Principle: Distribution of the distributive system permit to perform modifications a guarantee the robustness of the system in performing separated modifications. Industrial agents [102], permit to complement the distribution with intelligence.

Principle: Standardization some tools, service and different protocols that the system embraces during the system execution require well-defined standards that avoid compatibility issues on the reconfiguration. For example, interoperability standard protocols (OPC-UA, Modbus, MQTT) for sensors and other devices are some industrial communication examples for connecting these tools.

Besides support evolution, the standard principle is essential for the acceptance and convince of the industry of the benefits of such systems.

Requirement 3 (R3): The service reconfiguration must be performed according to the "nervousness" state of the system, *i.e.* reacting smoothly or dramatically in such a way that mitigates the problem [175].

This requirement is related to "what" to reconfigure level that is divided into physical and logical reconfigurations (see section 2.2.5). The type of modifications, likewise many other reconfiguration factors (*e.g.*, effort, profit, risk, and time) need to be evaluated. For a smooth reconfiguration is preferably small, and logical modifications rather than the physical modification. But it is essential to consider all types of modifications likewise many other factors re-configuration factors, (*i.e.*, effort, profit, risk and time). Besides to make use of the principle of evolution by taking small profitable reconfigurations steps, these are only taken if the system remains stable to evolve. Thus, a smooth reconfiguration that answers to the "what" to change must consider the following principles:

Principle: Cooperation - Interaction the adoption of a distributed service reconfiguration approach need to be coordinate to avoiding concurrent and conflict service reconfigurations that might lead the system to different directions. In this sense, it is also necessary to design well-defined collaborative interaction protocols.

Principle: Nervousness Control the distributed and autonomous entities are proactive to identify opportunities for adapting. In theoretical terms, whenever there are benefits, the system must be reconfigured. However, in manufacturing changes are never well accepted, particularly continuous changes. A rule-based solution allows you to regulate the number of reconfigurations that balancing the nervous system, and it is necessary to adapt these rules to a particular scenario.

To create a smooth reconfiguration process is necessary to consider the type of environment. If it is a non-real-time environment, it is possible to run a schedule to discover the most stable and smooth reconfiguration to be made. If it is in a quasi-real-time environment where decisions are reactive, the control of nervousness is consulted in order to stabilize the system for better evolution. Although introducing a higher complexity effort, permits to evolve smoothly respond easily to future disturbances.

Requirement 4 (R4): The service reconfiguration process should be performed in a distributed manner and consider competitive and collaborative scenarios [175].

The distributed service reconfiguration permits to build better and robust reconfigurations. However, the analysis and execution of the reconfiguration process are usually carried out individually without considering future impacts. Based on the entities' interactions, a feasible the reconfiguration can be accomplished, either in an individual mode or collaborative mode.

Principle: Collaborative - Interaction this principle promotes the collaboration between distributed and autonomous components, as a way to achieve better centralized or decentralized decisions.

The overall reconfiguration emerges from the local reconfigurations without joining in just one point the entire image of the system.

The collaborative interactions are implemented through negotiation strategies, supported by communication protocols that depend on the negotiation type used.

With the increasing development of reconfigurable industrial applications (particularly at the device level), the design and definition of requirements are essential. The requirements described in this section envisage reconfiguration steps that will direct as service reconfigurations with a direct impact on how industrial automation deployments will evolve.

2.6 Reconfiguration Requirements

Recent years have witnessed the rise of systems that attempt to recognize, simulate, or react to human emotions, with its roots traced to Rosalind Picard's 1995 MIT technical report [155].

2.6.1 Industrial Requirements

The use of requirements helps us to define the ideal solution and the characteristics of the ADvISER agents. Also, the analysis of these requirements allows foreseeing the type of solution is being done. As a result, the focus will be based on the following requirements:

Industrial Requirement 1 (*Distribution*)

Traditional control systems often operate in a centralized manner, upon rigid control and communication structures. In opposition to the traditional centralized structures, the new approach must be based on a decentralized control structure built upon a set of distributed entities, each one is independent and autonomous, with proper objectives, knowledge, and skills. An entity could be a production machine, a quality control station, the product itself or part of the information system. This means that the high-level functionalities (*e.g.*, control, scheduling, and monitoring) are distributed.

Industrial Requirement 2 (*Autonomy*)

The system operation based on the coordination of individual entities happens with a certain level of predefined autonomy, *i.e.*, with minimal assistance of external entities (*e.g.*, to an operator), or even without any intervention.

This process requires to clearly define the control of the entity, in terms of objectives, knowledge, and skills, and the ability to cooperate if necessary.

A manufacturing entity that exhibits some degree of autonomy is capable of response, from a local perspective, to achieve a goal. In other words, each manufacturing entity can act autonomously (*i.e.*, control their local behavior) in case of predictable and unpredictable environment circumstances. Examples of autonomy processes where autonomy can be present are change production parameters, like configurations of single production steps and sequence of production or testing steps, or change resource (*e.g.*, machine) utilization in a predefined range.

Industrial Requirement 3 (*Responsiveness*)

The designing of responsiveness mechanism in a manufacturing system represents the ability to adapt its tasks in a timely fashion, this characteristic is seen as a critical subject in the development of factories of the future. A very similar term is agility, which many authors state as an essential feature for the manufacturing system to be able to adapt to external changes in productive and cost-effective ways rapidly. Either way, these features allow responding to changes as recognized in Industrie 4.0 initiative quickly.

The levels of responsiveness and reconfigurability imposed by the factories of the future allow to better respond to changes, such as failures, by using distributed control structures. One way of looking at responsiveness to complete or adjust the assigned tasks within a given time is a two-step process: i) proactive reconfigurations and ii) reactive reconfigurations.

Industrial Requirement 3.1 (*Reactive*)

The reactive characteristic is an essential aspect in manufacturing control in terms of agility, in other words, it can respond dynamically and quickly to the disturbance. For this reason, a reactive action (*i.e.*, adaptation/reconfigurations) deals with stochastic processes without the ability to foreseen in detail the effects in long-term. Therefore this method should be used as a response to the lack of models that do not provide enough rigor. The continuous and reactive responses over the time allowed to make short-term changes and converge to the planned situation, even with limited accuracy.

Industrial Requirement 3.2 (*Proactive*)

In contrast to the (**IR 3.1 Reactive**) method, the proactive method considers taking some risks rather than reacting to them, as an example, instead of reacting to disturbances and deviations the objective is to use this information to improve the predictive model that triggers the proactive actions.

These methods are selected when effects of the actions/reconfigurations are based on consistent models that allow making predictions. The achieved results, in this case, are slowly, taking some time, but the planned impact model prove incredibly beneficial.

Even though it is desirable to act proactively throughout the entire time, it is not realistic. In highly dynamic environments, like manufacturing is crucial a continuous analysis of reactive reconfiguration strategy, meaning that prefers to deal with problems as they arise, and are not going out of the local control. A mechanism that relates the reactive proactivity method seems a suitable and necessary approach in the Industry 4.0 perspective. The prediction plays special attention that extends the traditional reaction mechanisms, in terms of minimizing the unpredictable effects and offers a global production optimization.

Industrial Requirement 4 (*Adaptation*)

The system must employ adaptation features to adapt the system behaviour with attention to self-adaptive and self-organization functionalities [173]. Any of these types of adaptation, in general terms, have similar objectives, that is, to increase the systems' flexibility, responsiveness, reconfigurability, and autonomy.

In brief, one way to explore the ability to self-adapt is accordingly to self-* properties that comprise the mechanism into four categories: i) Self-Configuring; ii) Self-Healing; iii) Self-Optimizing and iv) Self-Protecting.

Naturally, to support flexibility and reconfigurability, the system should provide the capacity to self-organize according to the current environmental conditions. In this field, many other adapting categories like self-* properties and some extensions fit the concept of a reconfigurable cyber-physical component.

Industrial Requirement 4.1 (*Self-Optimization*)

The system should exhibit self-optimization in order to improve overall utilization, performing self-diagnosis, and self-tuning its control parameters.

Industrial Requirement 4.2 (*Self-Adaptation*)

The system MUST provide the self-adaptation functionality Description: The system should dynamically adapt its behavior to a given physical production process state or in response to disturbances from both internal and external manufacturing environments. This functionality is related to the hardware and software reconfiguration.

Industrial Requirement 5 (*Learning*)

The system entities should provide the learning functionality Description: Aiming to improve the performance of the system (*e.g.*, productivity, products optimization).

Industrial Requirement 6 (*Cooperation*)

In distributed systems based on a community of individuals, entities must share their knowledge and skills/functions throughout the cooperation, in order to accomplish their functions (when they cannot perform them alone due to insufficient knowledge

or skills). Therefore, an interaction between the individuals is needed so that every entity can send /receive which functions it can offer and which are needed.

Industrial Requirement 7 (*Cooperation Interact with legacy systems and physical devices*)

Entities of the system must interact/cooperate with legacy systems and physical devices.

In addition to these aspects, there are some implications to consider when adopting agent technology, *e.g.*, the adoption is inseparable from some challenges like variability and unpredictability [4]. The introduction of flexibility in the system inevitably limits the ability to predict some future aspects with the same accuracy. While the distribution of decision-making processes undesirably introduces variability. Clearly, this represents a critical challenge that has to be mitigated, one possibility is to consider a more restrictive control and a process of control and coordination that limits variability and unpredictability [4].

2.6.2 Industrial Applications of Agent Systems - Existing Applications

Many researchers, during decades, has been discussing how Agent Technology is revolutionizing the industry, not only in the automation industry like factory process control, distributed sensing, industrial systems management [216] but also in other areas of application (*e.g.*, e-business, internet and telecommunications, robotics, logistics, aerospace, and air traffic control, and health care have also been reported in the literature, and particularly in several survey papers [103, 136, 149].

Given this background, it is known that the industrial automation industry imposes unique challenges requiring systems with a high degree of predictability, reactivity, and adaptability to meet the challenges such as unplanned production downtime or system performance deviations. All these issues can easily be reflected in revenue opportunities that are being lost. Agent Technology is one of the promising technologies that is often mentioned due to the multiple advantages in tackle complex problems.

Application in Manufacturing Automation

In essence, there are three major recognizable areas to use Agent Technology in manufacturing automation:

- During the first phase of engineering and **planning of automated systems**;
- Supporting **manufacturing operations** in automation systems;
- Lastly, supporting **validation and simulation**. During the evaluation phase of systems in which the functions of an automated manufacturing system are tested.

This shows that the adoption of agents in the industry is transversal to several areas. In fact, with all benefits accomplished to date in distinct areas, it becomes difficult to describe application domains in which this Agent methodology is not a useful concept when

designing all complex systems, it is easy to think of an agent-based architecture to solve the problem. This choice is often made because of the simplicity of the paradigm, being these the strong reasons that make Multi-agent Systems and Technologies so popular [217]. However, like any software, there are some circumstances for which it is particularly appropriate, and others for which it is not, for example in some cases it might not be accepted due to its inherent drawbacks like variability and unpredictability [4]. Naturally, we can find some contributions on when it is more appropriate to use this type of methodology and in which contexts [142], some authors even describe some common pitfalls in this situation [218].

When is an Agent-Based Solution is Suitable for Industry?

In areas of software development for automated manufacturing, there are a few aspects that we consider important and promising for industrial agent acceptance ([142] and [4]):

- **Inherent functional distribution** - This refers to applications consisting of delimitable sub-systems, each of which fulfills different tasks but has interdependencies with each other.
- **Structural alterability** - The possible interrelations between the system's elements cannot be fully specified (or only with great effort) at the time of development or are subject to particularly substantial changes at runtime.
- **Complex variable sequences** - The full determination of the system behaviour as a sequence of individual steps is not practicable or possible at the time of development; there is a need for flexible and dynamic adaptation of the sequences at the runtime.

These aspects are particularly relevant for the aptness of an agent-orientation paradigm. However, in spite choosing these properties appropriately along with the promising perspective of agent technology to solve the industrial requirements the truly deployment of agent for industrial applications are few illustrated in [105] and is far to be solved. As pointed out by [124] the technologically advanced for industrial applications founded on the Agent principles are rare. Usually, the deployment of agent technology **concepts in the industry is slow and sometimes restricted by the companies**.

Agent-based technology has proven to support agile and adaptive manufacturing control operations. This topic has been around for the past 20 years [24], having industry researchers, but more evident the academic researchers, expressing interest in its potential in manufacturing control architectures, and proposing many agent-oriented architectures as a way to model the holonic paradigm. Many of the proposed architectures have the holonic paradigm as a basis. The holon is composed of a physical part associated with a digital one [24], and can decide according to a certain degree of autonomy. PROSA [198], was the first reference architecture that was described in literature in 1998, other interesting architectures were proposed such as, CoBASA 2003 [8], PABADIS 2004 [120], ADACOR

2008 [106]), ORCA-FMS [146]. These well-known agent-based architectures exhibit benefits without significantly impacting production efficiency. More recently, ADACOR^2 2015 [196] and BIOSOARM 2016 [42] explored the introduction of bio-inspired principles for dynamic self-organisation. But then again they are still hard to be adopted by industry when compared to conventional approaches for industrial control. This question was investigated by many researchers on agent-based technologies, from the fundamental's theory [142]] to applied researcher, closed to corporations, [114, 124, 149] to understand the gap between the R&D projects produced on Universities/Laboratories and the research adopted in companies [114] states that the wider industrial adoption of agent technology involves the implementation of:

- **Factor 1:** Solutions are running in the industry that shows the maturity, flexibility, and robustness of the technology.
- **Factor 2:** Transparent reengineering process, showing that the implementation can be performed smoothly and compliant with the standards in the field.

These basic points are particularly relevant to the application of agent technology. Unfortunately, the right paradigm shift has not yet occurred, but many initiatives like the 4th technological revolution motivate companies to shift away from the traditional manufacturing infrastructure and embracing emerging technologies based on AI and ML [225]. This trend along with the many studies indicates that companies will invest in more in technology to motivate intelligent components and unlock new opportunities expecting higher productivity. This situation increases the importance of the work proposed here, leveraging opportunities that arise in several domains in the form of technological changes and paradigms, such as: Intelligent Manufacturing, Intelligent Manufacturing, Agile Manufacturing and CPS, which are being pointed as a suitable technology to develop systems that require adaptability, flexibility, robustness and last but not least the reconfiguration.

In fact, in these years, several projects put forward fairly recent paradigms as CPS and technology opportunities to persuade manufacturing companies to accept and implement Agent Technology.

2.7 Relation to the Current Work

Once the essential terminology has been explained and linked to the context of this thesis, it is now necessary to start reviewing the existing technologies under the Industry 4.0 vision. This analysis will highlight the different characteristics of the existing approaches, which will be fundamental for the design of an appropriate framework for dynamic reconfigurability. This is key step since building a framework that enhances dynamic service reconfiguration should not be a process started from scratch, but a continuous process where past approaches are pushed to answer research questions, including the ones defined in this thesis. A significant part of the research in service reconfiguration is devoted to changes in the service composition, aiming to compose the best service

that meets the client's requirements. This composition can be performed in two distinct moments: design-time and runtime.

Most of the works consider that reconfiguration is planned in the design phase, setting up the system to cope with expected changes [22, 34, 84]. In contrast, runtime reconfiguration reacts promptly to situations occurring during the system execution, to overcome unexpected events that were not considered at design-time (as illustrated by [37, 41, 197]). Either in design mode or in runtime mode reconfiguration must be carried out by choosing an appropriate action, or a set of actions (composition) from a range of possibilities.

The selection of different composition instants remains a challenge creating a design-time/runtime trade-off situation [228]. Particular scenarios result in complex computational problems that cannot be efficiently solved in runtime and, in such cases, a design-time policy might be beneficial in the long run, where time is not a constraint to apply more complex algorithms. Despite this unresolved challenge, many runtime reconfiguration approaches that operate in an automatic mode are explored, *e.g.*, by [41, 84, 97, 170, 171, 197]. The remain operation types, such as semi-automatic [37, 101, 144, 159, 163, 228] and manual intervention [22], are sensitive to the type of composition, which implies that the semi-automatic and manual kind of interventions shall be performed in the design time.

Regardless of the composition time, another important action comprehends the selection of alternative solutions that are available on the system. Traditionally, the discovery of appropriate services is performed in a centralized manner, *e.g.*, using the UDDI (Universal Description, Discovery, and Integration) central service registry [51]. Innovative solutions, *e.g.*, based on self-organization [9], refer to a cooperation process without any centralized decision, where a decentralization on the service discovery phase can be implemented by using the social plasticity of the providers, as presented in [37, 40, 84, 170, 197]. The structure of the new services composition is determined by considering a variety of techniques, from optimization techniques, which require heuristic-based algorithms to face the problem of combinatorial optimization (known to be NP-hard), to more sophisticated AI-based techniques, to achieve a near-optimal solution and to accelerate the execution time. Regarding the detection of the moment of change, several works are focusing on different reconfiguring conditions, *e.g.*, new consumer policies and requirements [11] or request of new services [126].

In the manufacturing domain, advanced data analytics aims to improve the understanding of unexpected behaviours and increase the processes efficiency [156, 221]. The key idea is to combine data from different sources, through the use of automated analytical tools and patterns recognition technologies, to generate self-reconfiguration opportunities and/or notifications capable of triggering actions by the human operators.

Industry 4.0 employs a wide range of ICT approaches including Cloud Computing and Edge, Internet of Things and Big Data Analytics, which have been explored for distinct strategies in manufacturing. Indeed, the number of publications using this technology for solving different manufacturing problems has increased (see Fig. 2.16 in the previous section, where the terms “data analysis” and “big data” appear more often). ICT advancements are used for many reasons, for instance, forecasting short-term product demand, for the automatic replacement of services based on the current production status, for condition-based maintenance and predictive analysis of malfunctions [12]. Within the domain of Industry 4.0, a very appealing technology is Cloud Computing that allows to quickly run complex data analysis, through ML algorithms, being extremely advantageous when developing decision systems for smart maintenance [12, 13], if the system has the flexibility of being reconfigured [25, 117].

In fact, this trend of Cloud-based applications for building predictive and predictive models is reflected in the increasing number of publications on this topic. For example for situations of identifying the trigger of adjustment in the context of maintenance scheduling [118], or to take advantage of parallel machine learning algorithms [221] for training large-scale of models. Recently digital twin arises in the context of Industry 4.0 based on increasing cloud technologies. The digital twin is a replica of a living or non-living physical entity allow to run *scenario* simulation based on computationally intensive models deployed on the Cloud to *explore* the behaviour of virtual prototypes in a what-if simulation mode under the control of the experimenter [122]. This type of solution will get even more important over the next years in describing the behaviour of the manufacturing system [81].

Currently, this type of trend is used to reduce defects through prevention and thus achieve higher performance and near-zero breakdowns in production machines (also called Zero-Defect Manufacturing). This method follows the principles of "fail and fix" maintenance practices to "predict and prevent". On the other hand, these systems require a lot of data, a training phase that takes place in design mode and active learning methods, which in general is modeled in a centralized way. Recent work reports the use of distributed multi-agent systems based on predictive models generated on the Cloud for the implementation of zero-defect manufacturing [153].

All this interest explains why companies commonly acquire powerful software applications, like SAP HANA, to successfully process and analyse large amounts of data in real-time. The use of services based on Cloud Computing solutions opens new business opportunities to deal with new analytic approaches which, although interesting, are less relevant under the scope of this thesis.

The focus of this work is in distributed intelligence systems, oftenly agent-based simulation are used to detect machine breakdowns, handling rush order arrivals, service performance degradation by using dynamic monitoring [19]. The specification of triggers during the design-time and using a common user interface was performed for manually added resource service's configurations [18].

In the manufacturing domain, service reconfiguration is also addressed in the literature. As examples, a services' model for the dynamic production system reconfiguration, particularly to reorganize the machinery to face a newly introduced product, was proposed in [123],and an approach that considers software and hardware reconfiguration is also proposed by [223], using a knowledge ontology and AI-planning for the service reconfiguration. The service reconfiguration can also benefit from integrating MAS and SOA to combine the best features of both worlds, namely decentralization, interoperability, loose coupling, intelligence and autonomy [166]. In this context, an automated service composition approach is proposed in [97], aiming to maximize the overall quality of the final composition, using agents that adapt the services' processes, continuously. A dynamic service reconfiguration is proposed by [197] that explores the use of agents to achieve consistent solutions focusing on fault-tolerant systems. An automatic plug'n'produce approach was used to quickly change the set-up services of a robot welding cell system to be more time and cost-efficient in small lot sizes [159]. In this work, the continuous service discovery plays an essential key to support the development of automatic reconfiguration in real-time.

In the context of several European R&D projects, the service reconfiguration problem has also been addressed, namely, the PRIME project [163] that uses a "plug & produce" approach combined with MAS, to allow the rapid reconfiguration and deployment, and the evolutionary system adaptation. The SOCRADES project focused on the reconfiguration of smart embedded devices [34], and the IDEAS project focused on the reconfigurable production systems by using agent technology to perform the online reconfiguration without the need of reprogramming efforts [144]. More recently, the PERFoRM project [67, 101] focused on the seamless reconfiguration of machinery and robots as a response to operational and business events.

The literature review is not clear in defining certain characteristics. Some projects are pointed out as having adaptability characteristics, but not all literature review report the same. This aspect with some divergence leads us to compile a list of some European R&D projects in Table 2.7, in accordance with our above definitions.

Table 2.7: Summary of the Literature Review on the Reconfigurability Applications of Manufacturing.

2006-2010 [32] FP6 - SOCRADES Service-oriented cross-layer infra-structure for distributed smart embedded systems	
Brief Description	**TRL Outcomes**
This project used the SOA at device and application levels to build a design, execution and management platform for innovative industrial automation systems. The project focused on designing and implementing a cross-layer infrastructure that would enable the integration of industrial automation systems and devices up to the MES/ERP level. The approach was driven by open standards, service-based integration, and collaboration among the various stakeholders, setting the stage for the next generation of automation systems.	TRL 3: Agent organisation model TRL 5: Cross-layer Service-Oriented Architecture On-the-fly reconfiguration methods Plug-and-Produce Devices and Systems Interfaces TRL 6: Reconfigurability automation systems TRL7: Composed automation services / Automation SaaS TRL 8: Atomic automation of Web services
2008-2012 [211] FP7 – FLEXA Advanced Flexible Automation Cell; Transport Domain	
The FLEXA project proposed to develop technologies, tools and methods for the validation of an automated, flexible cell, that can manufacture a generic process chain allowing for safe human interaction and deliver quality assured parts. The approach proposed was based on a web services architecture that connects the cell controller to ERP/MES. This project was validated with a case study in the domain of the aerospace industry.	TRL 3: Service-based architecture flexible routing of aerospace components Feedback and Feedforward quality control
2009-2012 [46] FP7 - FRAME Fast Ramp-up and Adaptive Manufacturing Environment	
The FRAME project had as main objective to shift the existing paradigm of the conventional system integration process and human-machine interaction to production system that are fully automated having capabilities of self-learning and self-aware. In this project a tool was developed capable of supporting the industrial ramp-up processes.	TRL 3: Ramp-up Time Reduction Strategies Self-aware and Self-learning Devices TRL 4: Semantic Information Backbone
2009-2013 [186] FP7- SELF-LEARNING Reliable self-learning production systems based on context-aware services	
The SELF-LEARNING project proposed to develop an innovative approach for enabling the integration of control and maintenance in the production systems through the development of self-learning solutions. This approach used concepts, such as lean and SOA principles for the development of integrated control and maintenance infrastructure. The results of the project were applied in three industrial case studies comprising machine tools, FMS and assembly lines.	TRL3: Self-Learning Middleware Distributed Energy Optimisation TRL4: Embedded Services Context Extractor for data driven model learning TRL5: Energy monitoring and optimisation strategies Security Service-Based Framework
2010-2013 [113, 176] FP7 - GRACE Integration of process and quality control using multi-agent technology	

The project developed, implemented and validated a cooperative MAS to integrate process control with quality control at local and global level. The MAS archi-tecture was designed to manage the planned changes of set-point in production processes and the large variety of unforeseen disturbances and changes in process parameters and variables. Self-adaptation procedures and optimization mechanisms for process and product parameters were implemented and integrated into control and diagnostic systems at local level, in terms of individual agents, and global level, considering the data gathered in all the production system.	TRL 4: Adaptation of functional testing plans and product customization Feedback control loops for online process parameter adjustments TRL 5: Multi-Agent System for integrating process and quality control Distributed data collection framework Self-adaptive quality control systems TRL 6: Multi-agent system infrastructure [102]
2010-2013 [144] FP7- IDEAS – Instantly Deployable Evolvable Assembly Systems	
The IDEAS project vision was to prove that the assembly equipment in industrial environments can be highly adaptable. The proposed approach for achieving this vision was performed through the implementation of a multi-agent control setup that could be reconfigured on-the-fly assuring the integration of different self-configured modules. The results were applied in a demonstrator assembly line, validating the proposed approach.	TRL 3: Skill-Based Plug-and-Produce TRL 4: Component-Based Engineering Tools Embedded Low Level Control Agent-Based Control Architecture On-the-fly reconfiguration methods
2012-2015 [163] FP7 - PRIME – Plug and Produce Intelligent Multi Agent Environment based in Standard Technology	
The PRIME project aimed at helping the SME integrating highly adaptive, reconfigurable, self-aware plug&produce assembly systems. The proposed approach was based on the use of multi-agent control, integrated monitoring, dynamic knowledge sharing and human-machine interaction. PRIMEs results allowed to create systems that are easier and faster to install with minimum costs and interruptions.	TRL 5 – 6: Integration approach for agent-based plug-and-produce systems Self-Awareness and Adaptation Agent-based Infrastructure Performance Monitoring
2010-2013 [133] FP7 - ManuCloud – Distributed Cloud product specification and supply chain manufacturing execution infrastructure	
The ManuCloud project had as the main goal was to develop an IT infrastructure based on the use of service-oriented principles. The proposed approach was based on the services of knowledge-based product, data exchange and quality assurance. The developed infrastructure can be applied in various application domains, for validation an industrial proof-of-concept case study was provided by Robert Bosch AG.	TRL 4 - 7: Intra-factory based on OPC-UA Intra-factory (cloud connector) Service aggregation and front-end side configuration
2012-2015 [74] FP7 - I-RAMP3 – Intelligent Network Devices for fast Ramp-up	

The I-RAMP3 project main idea was to empower the transformation of the traditional European manufacturing industry towards the implementation of smart manufacturing systems. The proposed work was based on network-enabled devices (NETDEVs), being these based on agents with self-descriptive capabilities.	TRL 5 – 6: Methods for Configuration & Optimization Plug&Produce devices with built-in intelligence Sensor and Actuator Middleware for rapid factory integration Global optimization models for automated device configuration
2010-2013 [84] FP7 – IMC-AESOP - ArchitecturE for Service-Oriented Process - Monitoring and Control	
The IMC-AESOP project investigated how to use a SOA approach for monitoring and control applications. The proposed architecture enabled cross-layer service-oriented collaboration both at horizontal and vertical levels by utilising service-oriented integration and the cloud. The feasibility of the approach was demonstrated on use cases provided by the end-users.	TRL 4: Architecture developed for ArchitecturE for Service-Oriented Process TRL 6: Cross-layer service-oriented architecture Monitoring and self-adaptation methods Real-Time web services
2012-2015 [125] FP7 - ARUM - Adaptive Production Management	
The ARUM project had as main objective was the mitigation of unexpected events during the production of complex, small-batch and highly customised products. ARUM's approach is based on a holonic MAS combined with a service-based architecture. The validation of the proposed approach was performed in three industrial testbeds, focusing on aircraft and ship industries.	TRL 4: Agent-based Infrastructure Component-Based Engineering Tools Service-Oriented Middleware TRL 5: Use cases [102]
2013-2016 [63] FP7 – ReBorn - Innovative Reuse of modular knowledge Based devices and technologies for Old, Renewed and New factories	
The ReBorn project proposed new strategies in order to support a new paradigm of re-use production equipment for rebuilding decommissioned production lines. The work was based on the use of self-aware and knowledge-based equipment. The results of this project were implemented for performing the re-design of factory layouts within various industrial demonstrations use cases.	TRL 4 - 5: Self-Aware Devices Life-cycle status self-assessment methods TRL 5 - 6: Device self-description Plug-and-Produce Component Integration
2013-2016 [185] FP7 – CassaMobile – Flexible Mini-Factory for local and customized production in a container	
The CassaMobile project intended to develop new manufacturing system within a 20'ISO container, with characteristics as mobility, flexibility and modularity, capable of producing highly customized parts. The system was built based ion mechanical and control systems adaptation of SOA systems. The developed work was demonstrated in three case studies, two related to the healthcare domain and another with the industrial area.	TRL 4 - 6: Modular mini-factory system System configuration using an Advanced Human Machine Interface (HMI) Task-driven adaptive automation Modular manufacturing system Plug & Produce systems
2013-2017 [184] FP7 – SelSus – Health Monitoring and Life-Long Capability Management for SELf-SUStaining Manufacturing Systems	

The SelSus project vision required the development of a new paradigm based on self-healing, self-diagnosed and self-aware production resources. This approach was developed based on web-based services for multi-modal data acquisition techniques to validate, update and document all information on the system health. The results were applied in two demonstration scenarios from two different industry domains, automotive and white goods	TRL 1 - 4: Modelling Language for Agent Knowledge Integration of Multi-Agent System, semantic and Modelling Languages TRL 3 - 6: Development of semantic enrichment of MAS Distributed Self-diagnosis models (TRL 3 - 4) TRL 4 – 5: Plug-and-produce communication infrastructure Could-based data persistence
2015-2018 PERFoRM H2020 – Production harmonized Reconfiguration of Flexible Robots and Machinery	
The PERFoRM project had as main goal the transformation of the conventional production systems into plug&produce production systems. For developing the proposed approach were taken into consideration several existing paradigms as CPS, service-based architectures and cloud services, among others. The results of the project were validated and deployed in four different industrial case studies.	TRL 4 - 6: Middleware & standard interfacing Technology Adaptors for plug and produce systems Reconfigurable and self-adaptive systems Visualisaton Methods to support Reconfigurability
2011-2014 EMC^2-Factory [127] 2011-2014 (EU FP7 NMP FoF): Eco Manufactured transportation means from Clean and Competitive Factory	
EMC^2-Factory objective was to develop a radically new paradigm for cost-effective, highly productive, energy-efficient and sustainable factories. It focuses on main energy-intensive processes within three industrial sectors in Europe (automotive, rail and aerospace), developing tangible and industry relevant results to be easily implemented in manufacturing environments.	TRL 4 - 5: Re-engineering of process as well as control, sensing and actuating technologies in terms of eco-efficiency Planning and control (*e.g.*, simulation) tools for eco-factories of the future
2016-2019 GO0D MAN H2020 – aGent Oriented Zero Defect Multi-stage mANufacturing	
The GO0D MAN project aimed to develop a framework in order to attain Zero-Defect Manufacturing (ZDM) when working in a multi-stage production line environment. The work was based on technologies as agent-based CPS, data analytics and integration. The final integration of the project results was performed in three different industrial use cases from a variety of industrial domains.	TRL 4 - 5: Feedback control loops for online process parameter adjustments Adaptation of functional testing plans and product customization Distributed data collection framework Self-adaptive quality control systems Multi-Agent System for integrating process and quality control
2015-2019 OpenMOS H2020 - Open dynamic Manufacturing Operating System for Smart Plug-and-Produce Automation Components	

openMOS aims to develop a common, openly accessible plug-and-produce (P&P) system platform which allows all stakeholders in the automation system value chain to collaboratively develop and exploit solutions. To achieve this, openMOS proposes integrating new P&P system concepts which have emerged in recent years, with well-established industrial-relevant technology platforms.	TRL 7 – 8: Open Manufacturing Operating System Instant Deployment and Change Over Methods Dynamic Energy Optimisation Methodology Standards for Device Self-Description and Interoperability Smart Plug-and-Produce Devices

Table 2.7 presents a brief description of each project, the year of the proposed solution, and also an estimation of the maturity of the research under development [56, 102]. The maturity of the research is represented by the Technology Readiness Level (TRL) measurement [54], which is a 1-to-9 measurement scale used to describe the level of maturity. By analysing the description along with the table we can recognise that a large part of the projects works with relevant industrial case studies, which is excellent for the projection of the works to go beyond the demonstration and validation phases. Table 2.7 gives the perspective on some TRLs practised by projects in the smart manufacturing field.

Table 2.8: Comparison of different reconfigurable applications in Manufacturing based on [67, 100, 102].

Year	Reference	Features											TRL
		Sevice Orientation	Agent	Standard Interfaces	Self-adaptive	Plug' n' Play	Responsivness		Requirement IR6		Human	Schedules and planners	
							IR 3.1 reactive	IR 3.2 proactive	Decentralization	Distribution			
2006 - 2010	FP6 - SOCRADES [32]	●	❍		❍	●							TRL 3, 5 - 8
2008 - 2012	FP7 - FLEXA	●		●							●		TRL 3
2009 - 2012	FRAME [46]				●					●	●		TRL 3 - 4
2009 - 2013	Self-Learning [186]	●			●								TRL 3 - 4
2010 - 2013	GRACE [176]		●	●	●		●	●	●	●			TRL 4 - 6
2010 - 2013	IDEAS [144]		●		●	●							TRL 3 - 4
2012 - 2015	PRIME [163]		●		●	●							TRL 5 - 6
2010 - 2013	MANUCLOUD [133]	●											TRL 4 - 7
2012 - 2015	I-RAMP3 [74]	●	●	●	❍	●			●	●			TRL 4 - 7
2010 - 2013	IMC-AESOP	●	❍		❍	●							TRL 4 - 6
2013 - 2015	ARUM	●	●										TRL 5
2013 - 2016	ReBorn [63]				●	●						●	TRL 3 - 5
2013 - 2016	CassaMobile [185]	●			●	●							TRL 4 - 6
2013 - 2017	SelSus [184]	❍	●	❍	●						●		TRL 7 - 8
2015 - 2018	PERFoRM	●	❍	●	●	●		❍		●	●	●	TRL 4 - 5
2011 - 2014	EMC^2-Factory [127]	❍		❍								●	TRL 4 - 5
2016 - 2019	Go0D MAN	❍	●	●	●		●	●	●	●	●		TRL 4 - 6
2010 - 2015	openMOS	●	●	●	●		●	●	●	●	●		TRL 7 - 8

Legend: ● - covered; ❍ - partially covered

Table 2.8 summarises and compares different reconfigurable applications in manufacturing, focusing on the desired characteristics defined for each main requirement. For each contribution, it can be easily confirmed if the requirements are met. If all the criteria are fulfilled, more easily a truly reconfiguration system will be adopted.

As already described, these features are particularly relevant to meet the projected requirements ensuring a good reconfiguration, but not only that, some are considered standards and best practices, allowing greater acceptance by the industry. For example for the data representation, XML, JSON, and RDF/XML are some of the prominent data formats for exchanging information, and these have a TRL of 9 [86]. For better interoperability and communication, SOA is a well-established standard and proven W3C with wide adoption across various domains as architecture and technology (e.g., SOAP or RESTful Web Services), having a TRL of 9 [86]. In the Industrial domain, SOA is reported with a readiness level 7 and sometimes 8 at the enterprise level. Nevertheless, if the adoption of Services is used at the Device and System level, along the life cycle of automation systems, these levels decrease to 5 - 6, as it is mentioned on [102], as it is still at an early development phase.

Observing the two Tables (Table 2.7 and 2.8) and focusing on the TRL, it can be seen that over the years the most significant focus is between readiness levels 4 and 7. This is not surprising since the objective of the program where the projects are included (Factories of the Future [55]) is covering the levels from technology validation in a laboratory environment (TRL 4) to the prototype demonstration phase in a real environment (TRL 7).

Moreover, on this compilation, it can be noticed a tendency toward the integration of MAS with SOA along with the list, mostly because SOA-based factory automation systems and Industrial software agents together can reach high levels of decentralisation and loose-coupling [175]. This integration of key technologies will leverage a much more flexible and effective way of equipment configuration, paving the way for the networked collaboration of software agents, potentiating production efficiency based on intelligent cooperation and easily reconfigurable manufacturing system.

Although there is no clear distinction of what constitutes CPS (*i.e.*, similar to the aforementioned Holons concept), this theoretical technology concept plays a vital role in intelligent and reconfigurable manufacturing, based on the simple association of "real/physical" and "cyber/virtual" world [12]. Even so, the coupling of cyber components and physical objects in automation is not new, but the CPS novelty consists on the interconnection of intelligent components via the internet [12], therefore the authors [175] believe that this might be a game change to increase the adoption of the agent-based technologies.

From a potential technology driver perspective, agents enable a certain degree of autonomous self-adaptative characteristics when integrated with SOA constitute a concrete expression of CPS (*e.g.*, the physical production equipment can be represented as Services, while the agents can be used for design control systems).

The essential point is that CPS is not pushing a new technology per se, but instead is combining existing technologies [104], then a necessary way of designing is to understand the design principles of this paradigm, in particular advocates key technologies that offer high levels of decentralisation and loose-coupling.

Nevertheless, from the designing perspective, the Industry 4.0 and the CPS principles require an entirely new way of thinking concerning the design of these systems [104], it is fundamental to create a plethora of procedures that guide system integrators. This lack of solutions and common guidelines to design this integration promotes a discussion inline with this thesis's RQs, of how engineers a reconfigurable system.

2.8 Summary

Different and innovative control architectures have been developed during the last years [21, 67, 144, 163, 176], but today manufacturers are still conservative in adopting modern approaches and moving from a conventional automation architecture to a decentralized [21]. Even though several groups are proposing how the next generation of automation system architectures should be, we still lack a set of roadmaps and techniques capable of supporting an effective and smooth migration to a modern factory were decentralized manufacturing technologies are enabled [67].

A literature study shows that, until now, the existing service reconfiguration approaches are usually specified during the design phase and do not consider the need for a smooth reconfiguration process (as stated by **R3**). Current approaches rely on a centralized composition planner that does not provide the capability to execute the on-the-fly reconfiguration (as stated in **R2**), being focused on the occurrence of failure events or product changeovers. The analysis and execution of the reconfiguration process are usually carried out individually without considering a collaborative analysis (as stated in **R4**).

Under this perspective, a crucial challenge in the service reconfiguration is to develop an integrated approach that addresses the established requirements, while considering the competitive and collaborative environments where they are applied. This should imply the detection of opportunities to evolve the system's behaviour, in a continuous and pro-active manner while elaborating and evaluating alternatives to complete the reconfiguration on-the-fly smoothly, *i.e.*, without the need to stop, re-program or restart the system.

In conclusion, and having laid out the background, including an overview of supporting concepts at the beginning of the chapter, we have identified gaps in the current literature and provided a survey of the existing definitions for reconfiguration design principles. In this process, we ended up developing our own definition, which we hope to be representative of an improvement over the existing ones.

CHAPTER 3

ADVISER: A DYNAMIC SERVICE RECONFIGURATION ARCHITECTURE

"Reconfiguration will require a while, but then the new form will make the computer much faster than the old setting..."

English Wiktionary

Reconfiguration is the beating heart of any agile and smart manufacturing system. Indeed, reconfiguration from a dynamic perspective cannot be reached without the ability to "play" with the behaviour of the system. This has prompted researchers' interest in new IT approaches over the last few years, with research being developed to simplify the necessary structural changes and behaviour of the system. However, the rapid advances, particularly in the manufacturing operations management's domain, made this topic even more complicated. This implies enabling the manufacturing software with the capacity to deal with unpredictable changes in an autonomous way.

The goal of this chapter is to introduce the main principles of a service reconfiguration architecture applied to manufacturing systems. Initially, the requirements for achieving reconfiguration are presented to assist in the design of the service reconfiguration approach. The proposed service reconfiguration architecture relies on the use of multi-agent systems, taking advantage of their inherent capabilities to facilitate the automatic monitoring, searching, selection and composing of tasks as the basic principles of the reconfiguration. Finally, we discuss different strategies to identify opportunities to reconfigure.

3.1 Initial Considerations

One of the most fundamental problems of a manufacturing control system is to control the operation plan, *i.e.*, make sure that the manufacturing operations are executed according to the scheduled plan. Unfortunately, the occurrence of production disturbances, like resource failures or urgent production requests, makes the system unpredictable, leading to variations in the system's performance.

As illustrated by Fig. 3.1, the requirements for reconfiguration strongly rely on the knowledge of the operator, for example, to decide when and how to adapt or evolve in a specific scenario.

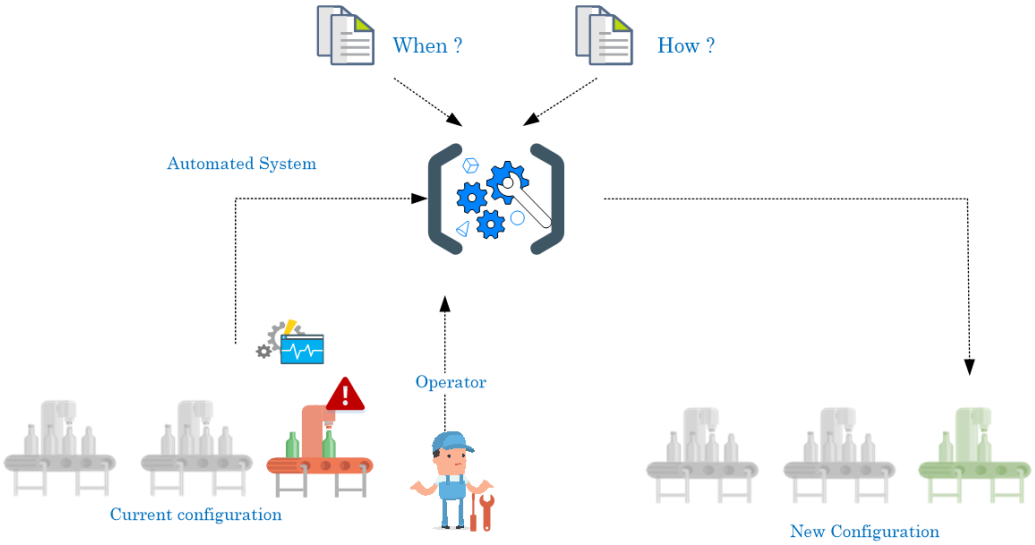

Figure 3.1: Overview of an automated system's requirements for reconfiguration.

Controlling such variables tends to imply systems that are capable of adapting or evolving in its features. However, such unpredictability creates a strongly dynamic environment, in which the events happen in an unpredictable way, complicating the selection of the action that better improves the system or mitigates a specific problem. Along with the unpredictability factor, there is still the issue of the decision time traditionally required.

Decision time should be as short as possible in order to moderate unpredictability and limit the "domino effect" and their possible consequences. On the other hand, rapid reactions may compromise the time necessary to find an optimal solution. In this context, the challenge may become even more complex, if combined with a dynamic environment, with increased changeability. Even if the reconfiguration decision solves the problem quickly, it may not be the most appropriate one since it may limit the flexibility of the system in the future. According to the **RQ1**, two sub-questions can be defined:

RQ1-a *In which cases is the reconfiguration likely to be implemented?*

The development of the reconfiguration process includes several activities, such as:

I. Facilitate efficient replacement during operation and enhance the overall effectiveness of the service to better return to use.
II. Guarantee that the reconfiguration objectives remain achievable in a long-term perspective.
III. Increase or, in the worst case, maintain the flexibility and reconfigurability of the production system, thus guaranteeing its evolution.

Regarding the above-mentioned activities on a reconfiguration system in an industrial factory approach, once the causes of the disturbance are identified, it is easier to limit the behaviour that the reconfiguration will have to do, in particular under what conditions it must be activated. These relevant questions are detailed in other sub-questions.

RQ1-b *What are the available options to detect the need for reconfiguration and how can such needs be mitigated?*

To achieve these objectives, the following activities must be considered:

I. Choosing reconfiguration triggers based on reconfiguration policy (when phase).
II. Select reconfiguration actions based on the reconfiguration mechanism (how phase).

The first point represents a model with the activation policies that traditionally is represented in the form of a "rules-based" system, ideal for quick responses to changes that usually happens in the low-level. This mechanism policy notifies when the potential reconfigurations are needed and can constitute an opportunity to improve the system's performance.

The second point introduces the reconfiguration mechanism as the core of the reconfiguration management system. The reconfiguration mechanism after identifying a reconfiguration opportunity should analyse the current configuration (*e.g.*, that which expresses the state of the system at a given instant) and selects actions that will guide the course of the system after. This includes a decision-making technique that combines the actual configurations with the desired strategies.

3.2 ADvISER Main Design Features

Traditionally, building reconfiguration applications in manufacturing cover many aspects (*e.g.*, static vs. dynamic; centralized vs. distributed; with or without human expertise input), which ends up introducing increasing levels of complexity during the design and development phases.

Thus, answering the *RQ1* is even more complicated when we consider applications that must be highly responsive and at the same time, continue to adapt and evolve. In short, there are some important software aspects that industrial factory systems must exhibit to support reconfiguration applications. They include the following *8-features*:

Feature 1: Provides consistency due to the Interface format, either for internally or using external interfaces. This means that the system can design functions across multiple systems improving the usability, there will be more standardization of functions and a clear and understandable format for software systems.

Feature 2: Contains AI algorithms to model reconfiguration opportunities in industrial environments supported by agent technology.

Feature 3: Introduces the integration of the services with agent technology to promote responsiveness and decentralization to build strategic profiles at runtime in a flexible autonomous and distributed environment.

Feature 4: Includes a service reconfiguration to build better strategic profiles for modelling, estimating triggering actions and also to limit the future generation strategies that need to be evaluated and addressed.

Feature 5: Includes a reconfiguration mechanism based on self-* properties under the service-oriented umbrella, which promotes the idea of evolving the services or systems with little effort by adjusting dynamic service reconfiguration.

Feature 6: Includes a reconfiguration triggering module to generate different triggers over time, depending on the strategy. This mechanism is responsible for determining

when it is suitable to propose a reconfiguration. In order to answer the questions when to reconfigure", the reconfiguration relies on a triggering strategy.

Feature 7: Determines *"how the reconfiguration"* can be implemented. The process consists in the elaboration of possible reconfiguration solutions towards the improvement of the system's utility.

Feature 8: The process of selecting the optimal system configuration in runtime requires an evaluation of the possible alternatives and a decision about the viability of its implementation, which is performed in the DRS module.

These features are essential for building a dynamic reconfiguration approach. Nevertheless, it is necessary to understand the existing control structures and how these resources can be an asset to deal with dynamic changes. In this sense, there are two types of modes, one simpler that follows optimal decision-making control, that is, centralizing control with a hierarchical structure. Alternatively, the other mode of control is based on its division by several distributed entities, *i.e.*, hierarchical structure, which allows reducing complexity and providing reactivity, but requires autonomous components.

As expected, each manufacturing structure has different characteristics. Therefore a hybrid structure that can temporarily change has been referred many times, but it turns out that balancing the trade-off between effectiveness and efficiency is far from being an easy task. Although the hybrid idea is fairly intuitive, in practice there are numerous challenges, such as the limitation of the prediction or even the complicated calculation of the trade-off (between effectiveness and efficiency), which is far from being effective, even more in variable and uncertain environments designed to cope with changes of production requirements.

With this in mind, after a look through the *8-features* that envision some insight on how these systems can be conceived, the next section proposes a generic dynamic architecture based upon the *7 Industrial Requirements*. The proposed distributed, multi-agent architecture aims to control the service reconfiguration management system using AI algorithms to learn best reconfiguration strategies and actions to approximate the optimal performance of the system in the short- and long-term.

3.3 Generic Design Framework

As previously presented, lately there has been increasing attention to design reconfigurable systems as a way to guarantee flexibility, agility, and robustness while dealing with industrial requirements. However, the efforts have been focused mainly on the points that reveal the clear benefits of using it, like the strengths of applying such reconfiguration and naturally the reconfiguration mechanisms (*e.g.*, mostly by applying heuristic techniques), putting aside the design principles (*i.e.*, the *8-features*). As such, the proposed Framework presents these aspects (see Fig. 3.2) to assist in the implementation of a proper reconfigurable system and demonstrate a better understanding of the presented architecture. These concepts show a set of basic functionality principles to define the processes involved in the manufacturing reconfiguration procedure.

The ADvISER Manufacturing design, represents the general flow of the traditional manufacturing process, illustrating the main steps of the workflow: from context diagnosis to execution and completing the cycle with performance analysis.

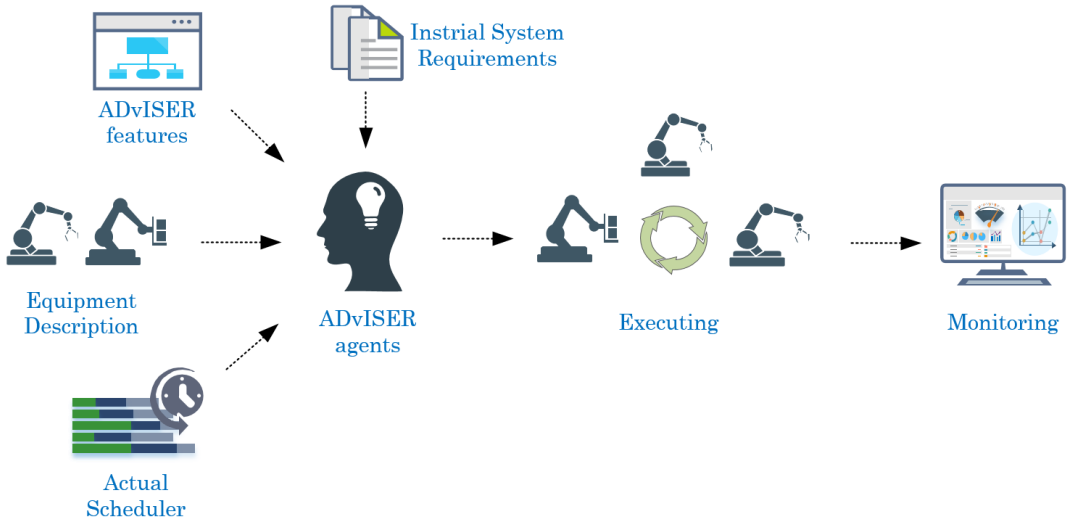

Figure 3.2: Overview of the ADvISER framework design.

Concretely, the control architecture designed for reconfiguration, which considers various input values, from the simplest ones like the *Description of the Equipment* to the more complex ones such as the *7-IR*, the *8-Features* and evidently, and evidently, the current description of the system in terms of *Actual Scheduling*. This vision gives ADvISER a good starting point for integrating the main components that make up the manufacturing structure. In order to properly deal with manufacturing control issues, including the reconfiguration management through a dedicated mechanism that handles the reconfiguration mechanism.

3.4 Reconfiguration Mechanism

In order to do any reconfiguration meaningful, we must know where we are going. Adaptations and improvements are two concepts that are spread across manufacturing topics and are often used in many contexts and situations. But how exactly do they work and what is the difference? In this section, we will look at the contrast between many reconfiguration types, and the most effective ways to use each.

3.4.1 Basic Reconfiguration Services

The idea of Service Reconfiguration in ADvISER is based on the concept of intelligent and collaborative agents. However, the means of how this has been achieved has not yet been described. Like in the self-organization principles, the system can adapt and evolve based on small changes, which might provoke more significant changes in the long-term. However, to achieve a long-term view, we need to understand the basic reconfiguration service actions that can be done.

At this point, the goal is not to achieve an overall understanding of a reconfiguration process, but rather to understand the basic steps developed for any kind of modification that will be the one used by each agent. To better understand it was presented some basic reconfiguration principles used for reconfiguration [208, 227], which comprehend three principles: changeability process, reconfiguration time, and triggers.

Changeability

Changeability can be seen in two phases: what we need to modify, and how to change it. Naturally, this relies on some changeability actions:

- *Reparameterization:* involves changing parameters of a component (*e.g.*, calibrating hardware parameters or functional parameters).
- *Rewiring:* intends to modify the communication structure between components (*i.e.*, creating or destroying connections).
- *Re-instantiation:* consists of adding or remove functions or components (*e.g.*, adding or removing nodes).
- *Relocation:* implies migrating functionalities among components.

The selection of these actions depends on the goal of the reconfiguration, *e.g.*, recovery, optimization, context change, ensuring the interoperability. In this way, to avoid lousy changeability decisions, we need to have a clear vision that the objectives of service reconfiguration are either to evolve the system or to adapt it.

Reconfiguration Phases

Depending on the type of reconfiguration strategy, namely evolution or adaptation, there are two essential modification phases with different characteristics, which are:

- *Design-time:* allows making modifications in offline mode, which is ideal for services that are evolving.
- *Dynamic:* avoids stopping the system, which means building a system that can be adaptable and run at the same time.

Mechanism Triggers

The mechanism's triggers consider in what conditions the adaption should be activated, this process implies questions regarding the objective of the modification:

- *Condition change event:* detect the occurrence of events related to the system, condition changes, *e.g.*, a resource failure or the removal of an existing resource/service.
- *Failure:* detect some violations of the service performance values, *e.g.*, QoS failure, lack of use.
- *Prediction:* evaluates the condition of equipment by performing periodic (offline) or continuous (online) equipment condition monitoring and predict interventions, *e.g.*, violations of the service performance values, *e.g.*, QoS failure, lack of use.

The primary reconfiguration cycle aims to solve many problems, *e.g.* to avoid major failures by making minor adaptations to improve the system. To satisfy these objectives, it is needed a mechanism that can generate plans to reach the original (*i.e.*, adapt), or new goals (*i.e.*, evolve). Ideally, the service reconfiguration process is conducted by the provider of the services, even though this activity is preferable to be supported in a fully automatic way, operators can start it as well.

In the ADvISER architecture, this activity belongs to the behaviour of the machines, which can use their autonomy or their GUI to force the reconfiguration. As indicated in Fig. 3.3, a quick overview of the reconfiguration cycle containing the idea of the reconfiguration mechanism can be seen.

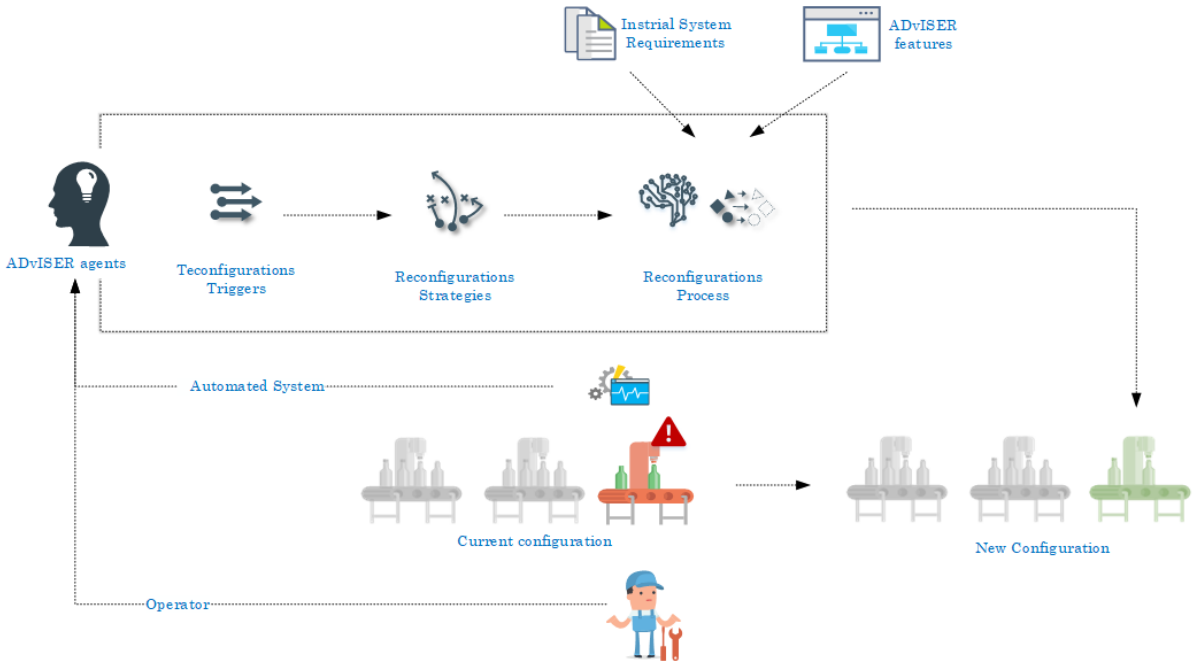

Figure 3.3: Overview of the ADvISER reconfiguration control mechanism.

In this overview it is possible to distinguish two important domains of influence, the automated system and the human operator. Fig. 3.3 consider the Human-in-the-Loop (HITL) for the ADviSER gather information from: (i) direct observations from the environment using sensors, (ii) registrations from the human operator's input.

The representation of the HITL combined with a distributed automated system in addition to enhancing the level of accuracy of the ADvISER, it permits to:

- *Reduce the margin of error of the reconfiguration*, which can avoid situations that damage production and the system itself.
- *Help to resolve new situations* that might appear, which are entirely new for the intelligent the system.
- *Help on the algorithm accuracy*, in the early stages, *e.g.*, when data amount of data available is irrelevant.

In this approach, two HITL roles can be involved. In the early stages, when the amount of data is not significant, the "knowledge experts" can assist with data labeling, tuning the ML algorithm. The human operator, on another hand, can intervene electronically (*e.g.*, use of dedicated dashboards or HMI) [35, 111]. Together, these inputs make ADvISER more productive and better efficiency, reasoning with more information which increases flexibility for efficient and quick reaction.

3.4.2 Reconfiguration Requisites

The developed system architecture for the service reconfiguration, in addition to strongly rely on the industrial requirements (IR) previously designed, it also integrates a broad spectrum of technologies, such as intelligent systems, machine learning, expert systems some of them defined in the IR, in order to supports the intelligent manufacturing system to adapt and evolve to face the volatile environment.

This framework can be used as the basis for engineering an industrial software solution to assist the production control based on intelligent control components, to regulate itself at opportune moments. This means that ADvISER framework is focused in to deliver high degree of predictability and self-reconfigurable for evolving in a smooth manner.

The functionalities of the framework and the separation of the responsibilities in the following modules:

- *Cyber-physical* components for interfacing industrial resources layer.
- *Data collection* where is performed the data acquisition from services that might represent different manufacturing sources (*i.e.*, specific services and properties of the services) is collected and upload to a database.
- *Knowledge management* is responsible to analyse the data based and run advanced data analytics models to help in the real-time analysis.
- *Reconfiguration module* that is responsible for the decision-making process resulting in actions to migrate to a new configuration, based on several reconfiguration strategies in this final stage is the visualization component.

3.4.2.1 Data Collection Module

This module is responsible for gathering and storing into a local database the relevant data for the system, such as functionalities of the components/devices in the environment like manufacturing events as introduction/removal/modification of hardware or software components in the physical context. Under this context, ADvISER considers two major sources of information, automated by direct observations from the different data sources from the environment or manually by the operator input, which requires to be electronic register (*e.g.*, use of dedicated dashboards or HMI). These inputs in the first stage can be used to generate knowledge based on analytical models (*e.g.*, monitoring rules to set threshold alarms and trigger automated actions).

3.4.2.2 Data Collection Prerequisites

Naturally, not all the data that is produced requires to be analysed, which means that collecting such amounts of information might not be useful in many situations. So, the following question arises: is it necessary to gather much information from many sources, as a Complex Event Processing (CEP) style? The quick answer would be "it depends". The first key in understanding is "clarity", how to make optimal decisions is based on the ability to gather information from a variety of sources and at the same time be able to synthesize this huge amount effectively. Often, gathering lots of data only confuses the analysis process. An essential concern in the data collection process is to make sure that information gathered is done in a way and for a purpose that is useful for the reconfiguration mechanism, and not jeopardize the system. For this purpose, some topics should be taken into consideration regarding the data collection process, such as:

- *Availability* of the data in terms of frequency. The reading of the sensors especially the frequency of data sampling, should be analysed, and the reading should be set to be useful for any reconfiguration.
- *Integrity* of the data is also imperative, the data collection process must ensure a read in a synchronous manner, preferable with the same without failures and with the same timestamp synchronization to ensure a good analysis.

– *Finally*, but not less important is data *security* and *confidentiality*. It is imperative particularly during the digitalization era, to keep the data on its machines protected against possible threats.

An important point that should be considered when developing such type of solutions is to reuse the widely used communication protocols and standards set by specific industry regulations, even more important with the increasing number of devices in smart factories (*e.g.*, machines, sensors or digitalized manufacturing processes) nowadays.

Developing this responsibility to the multi-agent system requires first to understanding how to collect the huge amount of data generated in Smart Factories and redirect the information.

3.4.2.3 Real-time Data

The ability to deliver accurate reconfiguration actions on time is key for any changeable system working in dynamic environments. Unlike traditional manufacturing, the modern manufacturing produces a huge amount of data about production, by joining the historical data and ML algorithms is possible to create computational models that can predict manufacturing interventions with accuracy almost instantaneously.

Literature reports several predictive maintenance applications focused on analyzing, in real-time, the generated data in order to identify failures before they occur and recommend the best moment to perform maintenance responses.

In ADvISER, in addition to preventing failures, we propose other strategies that require analyze the data instantaneously to improve the system while is running. Having this in mind, reconfigurations reactions need to be defined and built in such a way that enables response on time, improving production efficiency. Accordingly, in addition to accuracy, the responsiveness is another condition for data analytics to support quick reconfiguration actions.

This module will take advantage of real-time analysis to create extract knowledge from the data and deliver almost real-time accurate decisions. However, real-time analysis is not just processing of data at the time of its generation, it requires a set of tasks performed in advance, such as ETL process (Extract, Transform, Load) to transform the raw data for data mining and machine learning models, and then it is necessary to test and revise the models to ensure that they produce accurate predictions. After building a satisfactory analytical model, is set out pre-defined rules that are used as a performance indicator (KPIs) and saved on a database and ready to be applied in real-time production scenarios. During the real-time analysis, which works in a continuous loop, the data is processed and redirected based on two different reasons:

1. Redirect to the reconfiguration module. As will be explained further, the reconfiguration module converts the result of real-time analysis at first into self-maintenance actions to mitigate critical problems, only after is the moment to analyzed actions for system self-configuration action (e.g., self-optimization).
2. Display alerts in a dashboard, to assist the operator.

Today with the advent of ML algorithms and Big data technologies, data processing architectures has suddenly got a lot of attention. Typically, design and implement an application that can process

the data with a high level of accuracy and in almost real-time can be satisfied nowadays with two common data processing architectures such as Lambda architecture or Kappa architecture [93].

New technologies, specifically in manufacturing, play a significant role in digital transformation. The Internet of Things, 5G and Digital Twin, contribute to predicting the future with greater accuracy and if this computational power is implemented in cloud technology will be expected to increase the level of responsiveness, promoting thus a continuous improvement in the factory process. With the digitization of Industry 4.0, this process becomes closer and closer to reality.

3.4.2.4 Real-time Visualization Component

The results of the analysis (reconfiguration procedures) can be displayed almost in real-time to the operator through a **graphical dashboard** to potential assist in many possible reconfigurations like a maintenance task to return the system to its normal behaviour or triggers a maintenance warnings when an anomaly is detected in an earlier stage before the problem occurs avoiding machines breakdowns [111].

The design principles, to create user interfaces for data collection in industrial environments, should always take into consideration relevant features, such as the information presentation and the type of interface. The overall effects of the HMI appliance could not be totally predictable or even measurable since they do not depend only on the system design.

In brief, the general process of the reconfiguration system involves many stages, monitoring itself and its context, *detect* significant modifications, and *recommend* how to react, and by last, make appropriate decisions accordingly to the reconfiguration suggestion. All this process can be manually started by an operator input interruption or automatically by the agents. In brief, the architecture includes the modules that support the intelligent manufacturing system to adapt and evolve to face the volatile environment. The reconfiguration mechanism inside our agent consists of four parts:

1. An **environment-aware** module, that can indicate the conditions for reconfiguration, in relevant situations this can be an operator or an automated system;
2. A **reconfiguration trigger module**;
3. A **reconfiguration module** that lays down reconfiguration strategies;
4. A **reconfiguration process** that produces the navigation policy over the agent's actions to perform the reconfiguration.

Looking from a close perspective to each agent, the behaviour is a local loop to building continuous reconfiguration strategies. This approach assumes continuous supervision to regulate a specific catalog of services automatically. The agent focus moves continually across many phases (*i.e.*, observing, to analysing and enacting reconfiguration), which is set up on three components: **environment-aware**; **runtime reconfiguration module**, and finally the **executor reconfiguration process**. The proposed continuous loop approach for the service reconfiguration considers the use of MAS principles and intelligent algorithms to support two important phases. The when and how phases of the reconfiguration process, which is clearly relevant to mitigate the problem or evolve.

Naturally the majority of these reconfiguration processes belongs to the manufacturing resources type of agent, but generally, the reconfiguration control mechanism is set in all type of agents.

3.4.3 Reconfiguration Architecture Layers

Projecting a dynamic reconfiguration architecture is, in itself, a complex engineering problem if we consider a system capable of constant but different changes (such as, produce a new product, or equipment that has crashed, or lost their performance, or even an opportunity for improvement). Based on these requirements we use a multi-layer architecture as the baseline framework of our platform.

Each layer takes on different responsibilities, but as a whole the layered architecture can promote characteristics easily like flexibility or reusability of resources due to the dynamic characteristics. The dynamic connection among the layers permits to construct a complicated system from simpler components. From a manufacturing perspective, the workflow is dynamically configuring for a particular collective process to accomplish a manufacturing task that can change.

3.4.3.1 Service Layer

Having in mind the scope of the ADvISER in an industrial manufacturing automation context, all manufacturing elements follow the SOA technology principles and expose their abilities in the form of *Services* according to Feature 1. The use of services not only supports a simple interface for access to the physical manufacturing resources, but it also makes an easy integration from the agent-based reducing the complexity of the system. In fact, the device abstraction is important in many situations, e.g., in the presence of a large number of devices to either i) simplify the complexity of the device towards a clear selection or ii) integrate heterogeneous production components and interconnect them in a transparent manner covering the several ISA-95 automation levels.

The device abstraction development requires to develop a software-wrapper that allows to emulate the industrial device's functionalities and exposes it in the form of Services interfaces creating the typical *"Device-as-a-Service"* perspective. This clearly requires a semantic description and translation tasks using proper standard and appropriate protocols for communication with industrial legacy equipment (which is beyond the scope of this work).

Thanks to the efforts to promote a service-oriented paradigm in industrial environments, the companies are more agile and interoperable solutions, able to quickly and dynamically adjust their systems in an intra-layer and cross-layer perspective. However, invoke these manufacturing services, usually requires invoking more than one service to run a manufacturing process/routine. On this cross-layer interoperability, the service management gets a bit complicated because of two challenges.

- The first, the service involves two parties, *i.e.*, a provider and a consumer.
- The second difficulty is the correct coordination among the devices.

In addition to offering the services, the providers have to take care of the support during their service execution lifecycle. While the service consumer job is to invoke the services from the providers at runtime, taking into account changing conditions, constraints and goals. The result is a service orchestration activity that can be carried out by a human that assists in the modelling phase or if possible automatically. Thus, ensuring an efficient service reorganization performance at runtime and taking into account the constraints is difficult without an extra layer of intelligence (e.g., as defined in Feature 2).

3.4.3.2 Coordination Layer

Coordination layer is essential to specify, from the agent perspective, the relationship between agents and the services in order to promote the ability of a service orchestration process. Also, the ability to regulate industrial settings to achieve a reconfigurable manufacturing system. The ADvISER agents can assist in manufacturing operations in two situations:

- The agents can take over the production schedule, by decomposing the production order into smaller task orders and allocate, in an optimized way, to the best resources.

In some situations, (e.g., failures, breakdowns), requires a production plan or a schedule to be changed dynamically, which are hardly tackled by the traditional approaches [161]. The current reconfigurability and flexible requirements nowadays share the same concern, any situation that occurs must be appropriately reflected:

- Recognize the production configuration necessary to produce the product and identify the type of flexibility of the reconfiguration system together with the requirements.

In ADvISER, the idea is to go beyond this objective and to use the agents' ability and consider the existing industrial components as intelligent and autonomous entities. Even though numerous key characteristics of the agents are essential (*i.e.*, 7-IR), cooperation is essential for the agents to interact and negotiate the dynamic allocations between the capabilities (skills) of the resources according to the product constraints.

However, generating multiple plans requires continuous interaction between agents, which requires the need for automatic negotiation mechanisms to achieve a complete line configuration and task allocation. Of course, the challenge of planning changes during execution is more complicated. Planning during runtime will inherently create non-deterministic behaviour. On the other hand, the agent introduces greater adaptability in the system. In this context, this layer offers an extra level of intelligence in the service reconfiguration context, in the sense that:

- Each agent regulates a specific catalogue of services by discovering and select specific services, changing its catalogue of services;
- Each agent interacts with other agents to build a more complex composed workflow of services.

In this level, the service regulation performed by the agents is driven by the variation of the performance of the services.

3.4.3.3 Organizational Layer

The organization layer is associated with the global performance optimization and the design specifications, but in this case, it is not directly related to the services, but with the multi-agent perspective, in the sense that needs to deal with a high-level decision like new strategic requirements.

For instance, looking at Fig. 3.4, some of the requirements that will activate some sort of self-organization action in this level will be modifications at the *Description of the Equipment*, or in any of the *7-IR*, the *8-Features*.

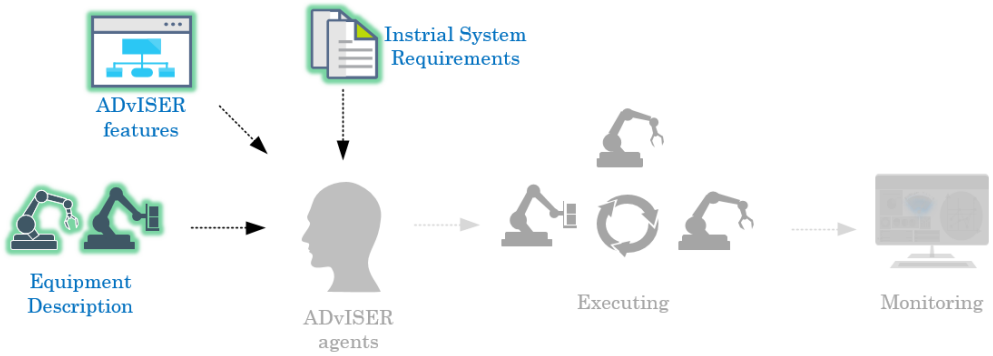

Figure 3.4: ADvISER Manufacturing.

Since the configuration of the manufacturing plant is evolving, the organization needs to be readjusted accordingly. The agents need to deal with the reconfiguration behaviour in a continuous form, trying to optimize the physical structural organization from a high-level perspective considering the new.

3.4.4 Key Point Summary

All these levels are decoupled from each other, providing a consistent platform for creating flexible and highly adaptive solutions. To make things even more interesting, we consider reconfiguration on ADvISER either in a design mode or during runtime. Some of the examples that can occur during execution include:

- Agents/Services entering the system;
- Existing Agents/Services leaving it;
- Agents/Services become unavailable due to any problem.

As a consequence, the ADvISER reconfiguration mechanism is forced to acquire general reconfiguration strategies to continuously self-organize the system, rather than depend on modifications that will stop the system. The agents need to realize the necessary action sequences per scenario: whether the strategy is to successfully cooperate or compete while remaining robust to any reconfiguration opportunity they might encounter in any of these cases. The effective control of the architecture lies in the reconfiguration mechanism of the agents and on the control flow between agents, which permit to reconfigure fast-changing requirements easily.

Thanks to a set of characteristics in production lines like agility, distribution or reconfiguration, modern manufacturing control systems are more flexible and capable of minimizing the effects of unexpected events. From the practical perspective, the architectures of these new manufacturing control systems must have the ability to adapt to changes in offline and online (*i.e.*, runtime) mode. To some extent, it is commonly accepted that changing the system while it's in operation (*i.e.*, in runtime) is far more challenging and involves a system that first evolves partially and secondly evolves dynamically. In these circumstances, the work detailed in the subsequent sections aims to provide the ADvISER architecture insights on how to deal with changes in runtime, like operations instructions during the manufacturing process, in particular, the operations for such adaptive manufacturing system architecture, along with the conceptual and practical perspective.

3.5 An Intelligent System for Production Control Systems

Intelligent manufacturing systems rely on their capacity to adapt and evolve to face volatile dynamic markets. This intelligent framework in Industry 4.0 is a new concept that will change the reality of how production operations are handled today.

We herein propose ADvISER that includes a framework and a production architecture that supports the creation of intelligent, distributed and easily reconfigurable manufacturing systems that, when applied to a Cyber-Physical ecosystem, will allow the regulation of the manufacturing operations most efficiently.

The ADvISER system will act at the MES level, interacting with the lower control level ("field level") which is related to the control of the automation devices, e.g., by using a network of Programmable Logic Controllers (PLC).

As it is suggested in [175, 176, 206], the idea behind the design methodologies present some limitations, namely they lack the possibility to present a holistic model of the execution environment to the developers, they do not deal directly with modelling techniques neither do they deal directly with implementation issues.

3.5.1 Methodology

Along these years, the agent community proposed methodologies to support the specification and engineering of software agent technology, such as AALAADIN [60] and probably the best-known GAIA [5, 220]. Even though these methodologies have their principles on object-oriented programming according to [82] and [5], these methodologies present some limitations, namely they lack in the possibility to present a holistic model of the execution environment to the developers, they not directly deal with modelling techniques and don't directly deal with implementation issues.

In this way, the methodology used to specify the ADvISER multi-agent architecture follow the one proposed in [5, 106], that comprise three main steps:

- The identification of the types of agents and their roles and functions (based on the analysis of the requirements elaborated in Task 1.1 of [5]).
- The specification of individual behaviours (by using a formal language, namely the Petri nets formalism that is suitable to model dynamic, concurrent behaviours).
- The specification of the interaction patterns and cooperation/coordination mechanisms (by using Unified Modelling Language (UML) sequence diagrams and communication diagrams) for modelling the overall behaviour of the multi-agent system that emerges from the interactions among its individuals.

The definition of the multi-agent architecture will follow the IEEE-FIPA (Foundation for Intelligent Physical Agents) specifications, which is a standard in the field of multi-agent systems.

3.5.2 ADvISER Multi-agent Architecture Vision

The proposed multi-agent collaborative architecture involves a society of distributed, autonomous and cooperative agents representing the components of the production system, including the products and the logical activities. Intelligent agents are acting autonomously on behalf of the

resources and representing logical entities, introducing intelligence and cooperating to achieve the global production objective, as illustrated in Fig. 3.5.

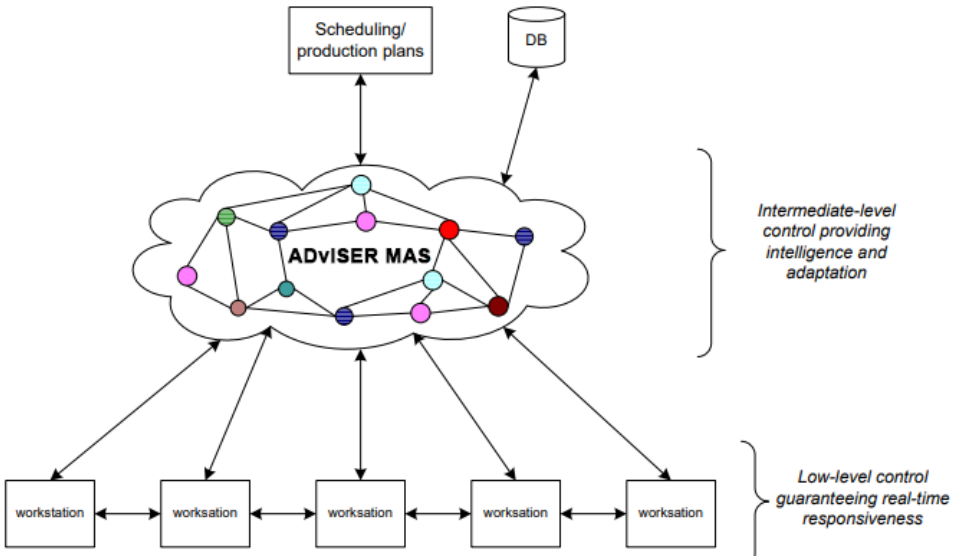

Figure 3.5: Scope of the ADvISER Multi-agent Architecture.

The whole production process will be supervised and will regulate the manufacturing operations through the networked collaboration of individual software agents, being the integration of the production and control processes sustained by the designed feedback control loops. In the following paragraphs it is detailed each characteristic of the generic agent.

3.5.2.1 Generic Structure of an ADvISER Agent

In the proposed society of agents forming ADvISER architecture, each autonomous agent (An) has a partial view of the system and behaves according to a small number of simple characteristics defined by the tuple:

$$An = \{G_i, P_i, B_i, I_i, S_i, A_i, K_i\}$$

where:

- G_i is the agent's objective, described by a mathematical expression to be maximized or minimized.
- P_i is a plan, *i.e.*, a sequence of actions leading to the realization of the goal.
- B_i is a set of behaviours to execute the agent's specific functions.
- I_i is a set of interfaces to connect the agent to its environment.
- S_i is the state of the agent, including the resources owned and its configuration.
- A_i is a set of attributes describing the agent's skills.
- K_i is the knowledge of the agent, consisting of a set of statements or facts.

Regardless of the type of resource of a typical production line (e.g., production, assembly or inspection control stations), all agents will be treated as intelligent agents, which means that agents either represent intelligent products being produced or other logical entities (e.g., with/without physical representation) the internal behaviour of the agents can be adapted and support the dynamic improvement. This aspect is crucial to support the reconfiguration.

3.5.2.2 Conceptual Structure of the Agent

In order to understand the conceptual structure of the ADvISER agent Fig. 3.6 illustrates the four main components: inter-agent communication, internal behaviour, internal database, and interfaces.

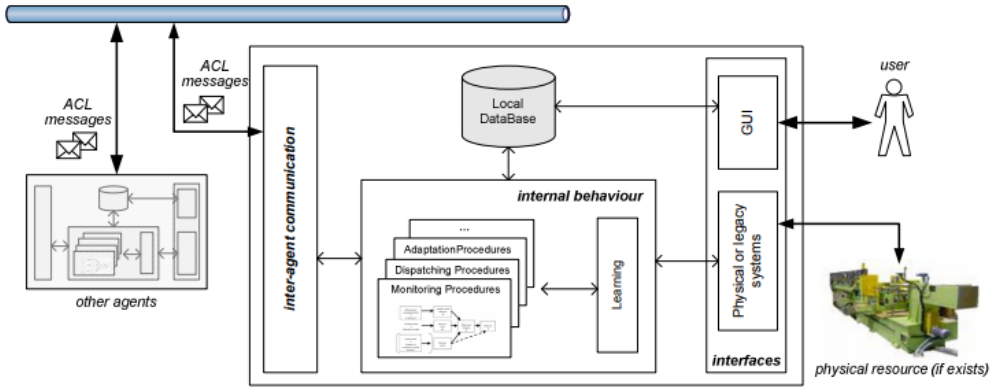

Figure 3.6: Conceptual Model for a Generic ADvISER agent (Source: based on [115]).

The requirements define the purpose of the agent, which consequently defines the type and, as a result, the internal behaviour of the agent. It is on this basis that different types of agents are designed.

In terms of functionalities, the designed agents, inheriting the multi-agent systems' principles, may exhibit a set of characteristics [107]:

- Modularity, *i.e.*, plugging in or out autonomous components/agents and also internal embedded algorithms.
- Adaptation, *i.e.*, applying local self-adaptive and self-organization concepts to adapt the system behaviour according to the environmental unplanned changes.
- Reconfiguration, *i.e.*, adding, removing or changing a component on the fly, *i.e.*, without the need to stop, (re)program and (re)initialise the other components.
- Responsiveness, *i.e.*, a better response to changes, such as failures, by using distributed control structures.

The agents may also possess learning capabilities to support the reconfiguration based on the knowledge gathered during its life-cycle. All these capabilities are crucial to support the adaptation in the control loops processes

3.5.2.3 Agent Interfaces

The idea of developing ADvISER is to manage and control a part of the system, simply without having to transfer information between different systems that are not integrated. Therefore, a technical solution attempts to implement in the ADvISER system an integration layer responsible for providing interfaces to be connected to third-party systems, e.g., ERP, MES, and physical equipment like robots and machines. As depicted in Fig. 3.7, the highlight layers reflect the appropriate interfaces that can be customized for each specific equipment/system available in the production system. All three different types of interfaces are considered in ADvISER multi-agent development, namely the integration with the physical equipment, the integration with the third-party systems and the graphical user interfaces.

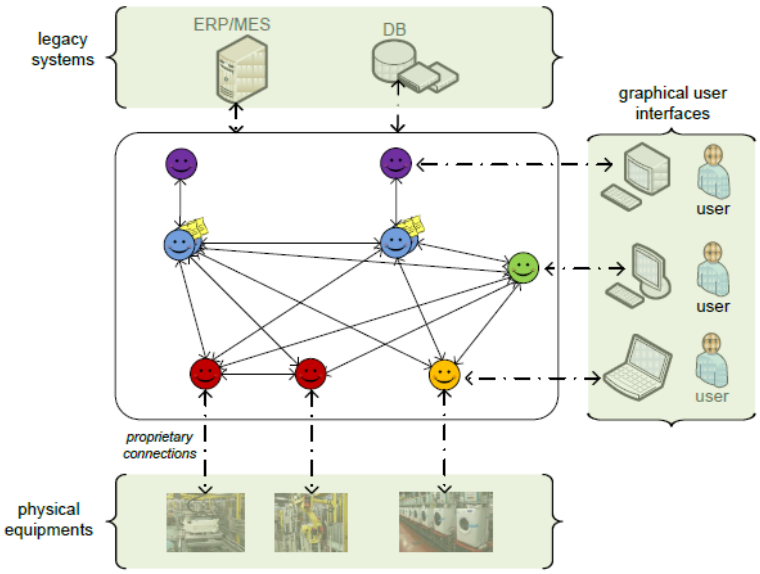

Figure 3.7: Representation of all types of Interfaces in the ADvISER architecture.

Taking into account the characteristics of the various interfaces, it is possible to generalize two basic types of interfaces:

- **Inter-agent communication**, it is responsible for the interaction with the other agents, making transparent the data exchanged to support the cooperation among the agents. In the inter-agent interaction of the ADvISER, the agents will follow the IEEE-FIPA specifications as mentioned above.
- **Intra-agent communication** is responsible for supporting the interaction of the agent with the physical equipment (e.g., robots and machines) and/or legacy systems (e.g., MES systems) if they exist and finally with the presentation layer.

From the technology point of view, these types of communications are accomplished by defining service interfaces from the service layer, which permits the ADvISER architecture to be connected either to the old legacy systems and also with the new generation intelligent manufacturing. Ideally, the interfaces can be customized to each particular system/equipment available in the production system.

3.5.2.4 Interfaces between Agent and Algorithms

In the development of ADvISER, it was formerly defined the possibility of using several types of services, with different purposes, but very appropriate for reducing the complexity of the agent, a generic example is shown in Fig. 3.8.

Figure 3.8 shows an agent, in this case, a Resource Agent, with the ability to access different services, either the services that the agent itself offers as well as external services that help to encapsulate complex functions. The complexities of these functions are performed outside the agent cycle, which permit to take advantage of the techniques investigated so far, in particular service-oriented, where the idea is that the agent is able to reuse tasks offered by specific APIs' interfaces, which provide certain algorithms quite optimized. Some of these services may reside on certain servers, which allows sharing the agent's computational load in specific contexts. For this

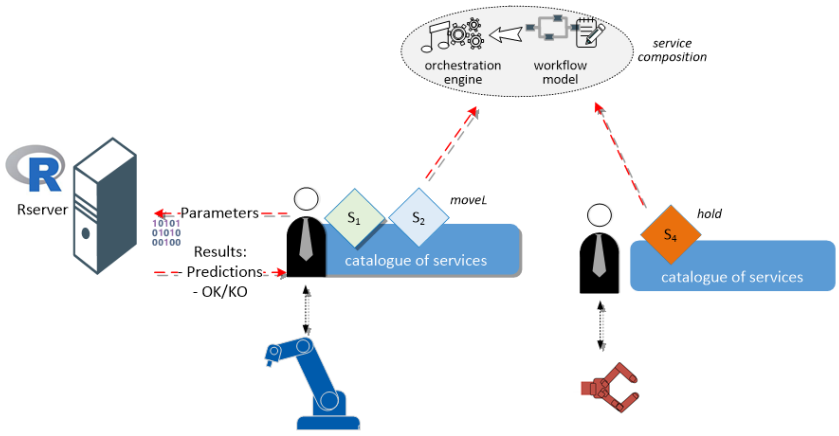

Figure 3.8: Interfaces in the ADvISER system.

and from a technological perspective, several interfaces must be developed in the development of the system.

3.6 Emergent Behaviour of the ADvISER Systems

The basic idea behind ADvISER architecture is to support the automatic reorganization of the manufacturing resources through dynamic control. As its name suggests, ADvISER (*A Dynamic Service Reconfiguration with Multi-Agent System Architecture*) promotes a suitable dynamic reconfiguration of services based on a multi-agent system oriented to services, which follows the Industry 4.0 vision, particularly for continuous manufacturing reconfiguration.

A general idea of the distinct agents that constitute ADvISER manufacturing framework is depicted in Fig. 3.9, in which the agents can represent manufacturing resources and processes.

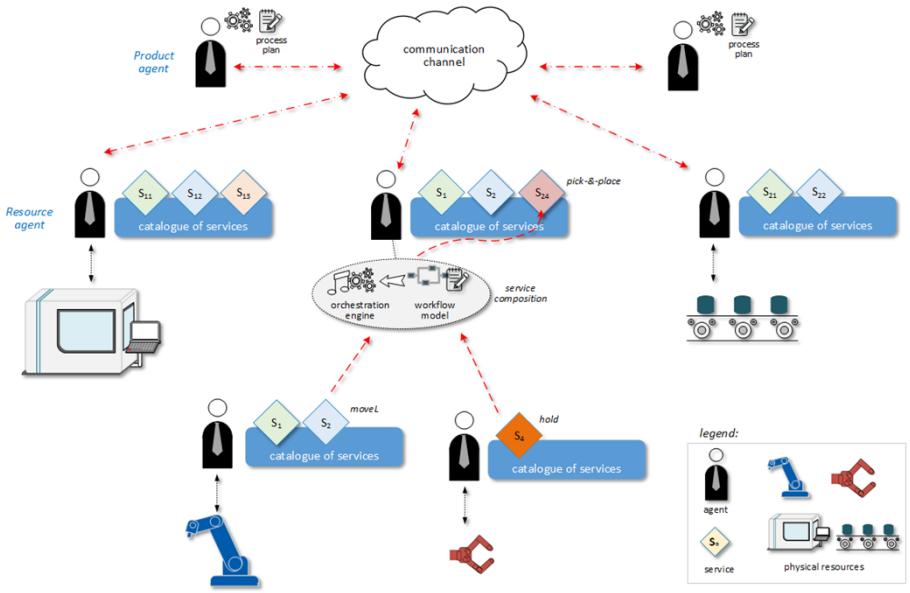

Figure 3.9: Multi-agent system approach for the distributed, dynamic and on-the-fly service reconfiguration.

The architecture defines two different components that cover the ADvISER's vision in the design of a distributed dynamic control: *services* and *agent technology* — considering the scope of the work, for the most part in the smart manufacturing system, which has the particularity of the product having the capacity to drive its own production (*i.e.*, intelligent product).

3.6.1 Intelligent Product, a Cyber-Physical System View

The concept of intelligent product [110, 128, 132] is a new industrial manufacturing control paradigm, aligned with CPS. Intelligent products carry the knowledge about their characteristics, wirelessly connected to share, in real-time, information about their state or environment, or to communicate with other cooperative objects (e.g., IoT) [150]. In addition, intelligent products collect and store data to support the implementation of monitoring, traceability and decision-making functions. The designing of ADvISER under this intelligent manufacturing perspective add important benefits, namely:

- Establishment of a product-driven production approach (*i.e.* the product takes the initiative during the plan execution [207]).
- Improvement of the entire life-cycle of the product, comprising the design, production, distribution, operation and end of life [148] phases.
- Improvement of the product quality and performance through the application of self-* properties.
- Improvement of the next generation of the product.

This strategy constitutes a distributed artificial intelligence solution, each agent possesses its knowledge and skills, being that the intelligent global system behaviour emerged from the interaction among the distributed agents. Following the principle of "divide to conquer", the ADvISER multi-agent system replace the centralized control by a decentralized functioning, allowing a high degree of flexibility, robustness, and responsiveness, which are not provided by centralized solutions.

3.6.2 Types of Agents

The ADvISER agents can recognize and represent the industrial resources that exist in manufacturing (e.g., products, machines, conveyors, and operations, by pre-engenering the equipment description). Naturally, the behaviour of the agent is designed considering the type of resource and the roles that it will act upon. Another ability is the mental model of the agents that reflects the different systems' model, which impact the services provided by the agents. Given this, we design various types of agents. Formally the agent's ecosystem is represented as:

$$Ecosystem = \{A_1, A_2, \ldots, A_n\}$$

Each autonomous agent (A_n) manages the services for different purposes, in different manners depending on the type and roles of the agent (see Fig. 3.5).n such a distributed ecosystem, the proposed architecture identifies several types of agents, according to the process of ADvISER. The set of agents in ADvISER can be represented by:

$$\textbf{ADvISER} = \{RA_n, PA_n, PTA_n, Rapporteur_n\}$$

Figure 3.10 comprises four basic agent types [107, 175] represented in the UML Class diagram represented containing the functions associated with each type of agent.

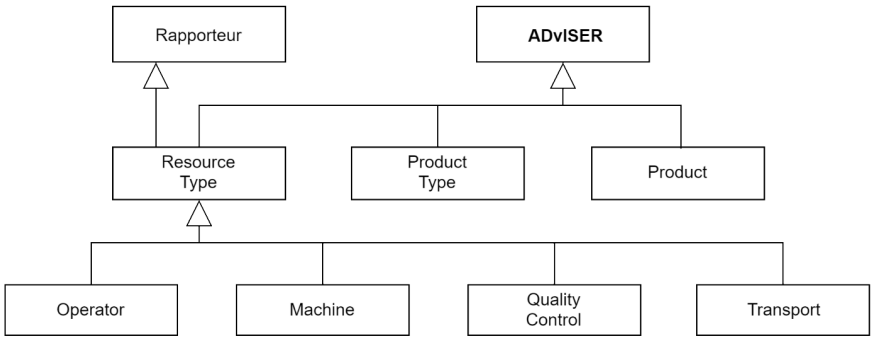

Figure 3.10: Class Diagram for the Identified Agent Classes.

Next, each type of agent will be described, namely its roles, functions, and behaviours.

The generic class of **ADvISER agent** contains the common functions that will be inherited to the other classes. This strategy guarantees identical reconfiguration functions in all classes of agents. As depicted in Fig. 3.10, the reconfiguration function relies on a small set of self-* properties (see the entire list in [94]) defined as functions. This very important concept brings the reconfiguring ability to the overall architecture:

- **Self-configuration:** capabilities can make actions that permit to automatically reconfigure themselves, for example for (re-)setting some services values, configure or re-organize its internal behavior with the purpose to affect the set of services. This mechanism involves actions that are the core of changes (what to change) question.
- **Self-healing:** The agents by adopting the self-healing competences, gain the ability to recover and adapt from undesired situations in two steps. The first step considers the procedure to detect disturbances while the second step works as a parallel process that will react to mitigate disturbances. Naturally, this process involves an earlier knowledge of the system concerning the ability to detect symptoms and how to provide alternative solutions.
- **Self-optimizing:** An essential goal of self-optimization is the ability to improve the efficiency (e.g., using the inputs in a "right" way) and effectiveness (e.g., provides the desired output) of the system. This process involves situations that can be performed reactively (*i.e.*, maintain the various operations updated) or proactive, (*i.e.*, plan ahead several optimization outcomes, through proactive tests and making every effort to optimize the operations).
- **Self-protecting:** The agents, by adopting the self-protecting competences gain the ability to anticipate and react in order to keep the integrity of the system. This requires several types of actions in order to protect itself and consequently the system from harmful services, for example, avoiding inadequate services, or even turning off unnecessary services.

Agents adopt all these capabilities (*i.e.*, self-* properties) to help to develop an autonomous, predictable, and responsive system to various changes. However, it is necessary to be aware of the use of the mechanisms. That is, although they are independent of each other, they may be complicated to handle if all are executed at the same time. It is essential to understand that there are mechanisms that affect others with their activation. For example, performing self-healing is going to have an impact on self-optimizing. To avoid this scenario, the final choice will be left

to the agent, in the reconfiguration control mechanism, which will be performed based on the criterion of adaption or evolution purposes. The responsibilities of both possibilities require the understanding of the reasons and steps to adapt, which serves as a bridge to connect the problem to be solved with the most sensible resolution of the moment.
From a practical perspective, the number of agents present in the system is strongly related to its characteristics. In the following sub-sections, all types of agents will be deeply described.

3.6.2.1 Product Type Agent

Product Type Agent (PTA_n), representing the catalogue of products that can be produced by the production line. They possess the knowledge required to produce the products [165].

These agents' type has the following responsibility:
- *Manages the catalogue of products* that can be produced in the production line.
- *Launches* the requested **batches of orders** to be executed in the production line according to the production orders coming from MES.
- *Traceability* of the **produced products** in the production line.
- *Monitoring the historical product execution* of a specific type of product (that can be used to apply traceability procedures).
- *Adaptation* (*i.e.*, self-* properties such as -optimizing and -adjustment) **of the process plan** associated to the product model based on the feedback of previous executions (e.g., adaption the product-change over time).

In Table 3.5 is a representation of the properties of the PTA agent.

Table 3.1: Summary of the Product Type agents' properties.

ISA-95 level	Communication layer	Properties	Mandatory
L2-L4	FIPA specification; Services;	Autonomy; Cooperativeness; Proactive behaviour;	Yes

This type of agent will operate at a higher layer than the others. The final number of product type agents will be defined during the deployment phase by the family variety of products being produced in the factory plant.

3.6.2.2 Resource Agent

Resource Agent (RA_n), represents the physical resources of the production line, such as robots, machines, quality control stations and operators that are responsible for managing the execution of the operations along the production line [165]. Under this perspective, a variety of specializations of the Resource Agent will be specified, but all will have some common responsibilities:
- *Expose the functionalities* of the resource as services, by encapsulating resources' functionalities and publish the services that they can offer (*i.e.*, each resource acts as a services provider).
- *Execution of services* related to the production operation processes.
- *Collection and storage* of the service execution's performances related to the invocation of the processes in the resource in real-time and based on the historical data (e.g., inspection tools data, or even human-machine interfaces).

- *Monitoring the evolution* of the resource performance of each service executed to observe opportunities to reconfigure (e.g., possible Service QoS degradation).
- *Provide feedback* about the invocation of the manufacturing service to the intelligent product agent.
- *Conduct the service self-reconfiguration loop* based on the historical data collected (that includes historical data and real-time).
- *Proactively* planning-ahead to identify and prevent problems.
- *Dynamic reactions* to unexpected service situations.
- *Coordinate* with other resources in a collaborative environment.

Aiming to perform a catalogue of operations, each RA possesses a list of skills representing the resource's capabilities, namely the list of tools, speed and energy consumed. In Table 3.2 is a representation of the properties of the RA agent.

Table 3.2: Summary of the Resource agents properties.

ISA-95 level	Communication layer	Properties	Mandatory
L1-L3	FIPA specification; Services;	Autonomy; Reactiveness; Cooperativeness; Proactive behaviour;	Yes

Contrary to the previous example, the idea of these types of agents is strongly linked to reactivity, so their operating range is between L1 and L3.

All these properties will be inherited by the other specializations of the RA, namely:

- *Quality control agents (QCA)* - manage the execution of quality control operations over the parts being manufactured. Particularly, they are responsible for adjusting the quality control algorithms and parameters, based on the environmental conditions and information about previously executed operations.
- *Transport agents (TA)* - represent the devices performing transportation operations, namely Automated Guided Vehicles (AGVs) or conveyors, transporting/routing the products identified by their product agents from one manufacturing station.
- *Operator agents (OA)* - represent the manual operations performed by operators. Using (HMI) human-machine interface to support the human input, such as visual inspection.

The number of instances of the Type RA will be defined during the deployment phase of the new assembly line.

3.6.2.3 Product Agent

Product Agent (PA_n), handles the production of product instances production line. The PA as Intelligent Product can decide the way the product is being produced, according to a process plan that specifies how to produce the product, sometimes referred as "on-demand production" [110, 165]. Figure 3.11 for instance, illustrates a specific production scheduling set that was already given at the design time, or the agent needs to discover either in design or runtime.

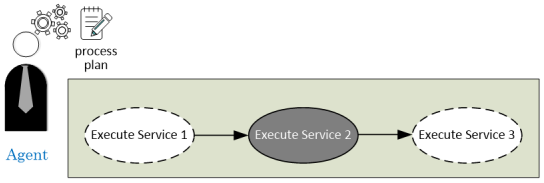

Figure 3.11: Overview of the PA following a specific process plan. Adapting global policies for the system.

The objective is to create one PA by each product instance with the following main functions along its life-cycle:

- *Collection of the production data* along the production line about the execution of the product.
- *Monitoring of the traceability* and data analysis of the product along the line to perceive deviations from the plan.
- *Management of the production* process of the **product** by interacting with resources to coordinate their actions according to the production plan.
- *Adaptation of the process* to be executed by the resources (e.g., an adaptation of inspection service parameters of the operations).
- *Drive the product* (e.g., re-routing) to face the current situation of the production process.
- *Creating a trust-based model* based on the resources (e.g., drive the product to the most reliable resource).
- *Proactively* planning-ahead to identify and prevent problems.

PAs can run on the cloud, particularly using cloud services environments, e.g., High Processing Computing (HPC) environments to achieve faster processing capabilities and can also dynamically self-organize in different clusters. Table 3.3 is a representation of the properties of the PA.

Table 3.3: Summary of the Product agents properties.

ISA-95 level	Communication layer	Properties	Mandatory
L1-L3	FIPA specification; Services;	Autonomy; Reactiveness; Cooperativeness; Proactive behaviour;	Yes

The Product Agent is also another type of agent working at a low ISA-95 layer level (e.g., Intelligent product at [110]). The number of instances of the type PA will be defined during the runtime phase of the assembly line by the product type agent that is strongly dependent on the daily production orders coming from the MES.

3.6.2.4 Rapporteur

Rapporteur ($Rapporteur_n$), is responsible for several tasks, such as coordinate entities to avoid non-beneficial reconfiguration situations with more precision, triggering adaptions on systems based on global monitoring and optimized global operation strategies.

The rapporteur enforces a regulation environment by introducing a kind of hierarchy in the decentralized system allowing to implement and achieve better optimization and data correlation. The rapporteur enforces a regulation environment, based on:

- *Collection of the distributed production data* along the production line about resources.
- *Reasoning* about the **best configuration** to avoid deadlock situations when the resources are automatically adapting (e.g., simultaneous reconfigurations).
- *Proactively planning-ahead* to identify and prevent problems.
- *Propagation* of **new solutions** to suggest the resources chosen for a potential reconfiguration.

In Table 3.4 is a representation of the properties of the *Rapporteur* agent.

Table 3.4: Summary of the *Rapporteur* agents' properties.

ISA-95 level	Communication layer	Properties	Mandatory
L1-L3	FIPA specification; Services;	Autonomy; Cooperativeness; Proactive behaviour;	No

Perhaps of all the agents, the Rapporteur will have the least direct impact (hence the mandatory as "No"). It is important to maintain consistency and to leave the same communication for the various ADvISER agents. The objective is to create one agent **Rapporteur** by each ADvISER instance. The creation of more agents of this type can be also be achieved if there is a bottleneck problem, by using only one.

3.6.3 Interaction Patterns

The resulting multi-agent architecture is based on a set of individual autonomous and collaborative agents, each one with its own objectives and behaviours, possessing its own perceptive and cognitive capabilities.

Since each one has only partial knowledge of the problem, they need to interact with each other to achieve their global objectives. Analyzing Fig. 3.12, it is possible to verify the existence of two separated layers [165]:

1. The higher level represents the ADvISER multi-agent system layer that is responsible for providing intelligence and adaptation to the system.
2. The lower level represents the physical control that is accessed by the RA services.

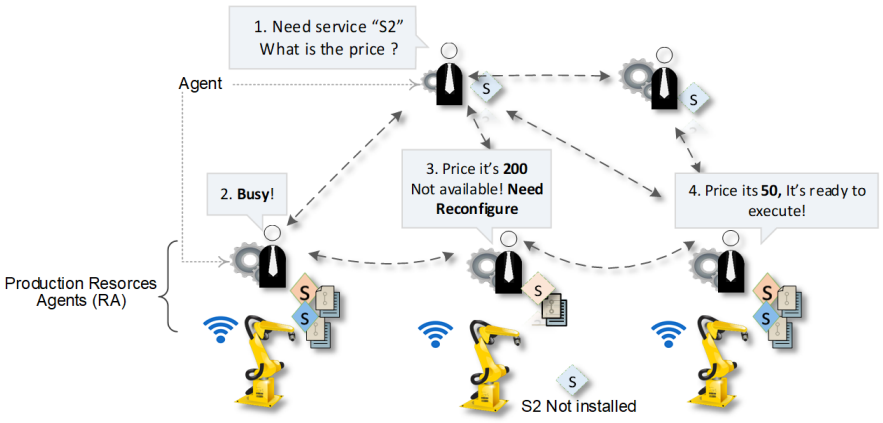

Figure 3.12: Multi-agent based Service Reconfiguration.

Besides this, interaction patterns among agents are designed to enhance integration and collaboration among intelligent units, to perform the reconfiguration, adaptation, and optimization in the production control, allowing the achievement of a global modular, distributed, adaptive and reconfigurable control platform.

At this point, it is necessary to understand the type of behaviour on the part of each agent and the iterations between them. In fact, it is in this way that the allocation of tasks (*i.e.*, manufacturing scheduling) to specific machines is accomplished. The service consumers (such as intelligent products) need to consume the production resources' services to meet the production demand, for this, the agents need to reach an agreement

The allocation is performed by a negotiation protocol with multiple rounds based on many features of the RAs like the local schedule, services performance/availability at a specific price. Analysing Fig. 3.12, the RA agent publishes the processes as services (*i.e.*, they act as service providers). These services requested by the PA permit to produce the product. If the product complexity requested by any PA increases, also the cooperation among the RA agents to compose a set of series to reach a solution. The cooperation to achieve a feasible composition requires the RA agents to use specific tools. An example is self-discovering mechanisms to find possible services relevant to the PA (consumer), *i.e.*, the agent can reason about the service's skills (e.g., trying to create possible coalitions between services). The idea is to adapt its catalogue of services towards the best services configuration to provide, using the service selection mechanisms to the analysis of the performance of the composite workflow. From another side, the PA pretends to discover the best services to create a better product, which requires implementing adaptable agent behaviours.

Let us take a look at Table 3.5, which represents the necessary interactions between the different type of agents previously described. The agents follow three types of interactions, *i.e.*, FIPA Request, FIPA Contract Net, and finally, ADvISER interaction pattern.

Table 3.5: Agent Interaction Summary.

Init\RESP	PTA	RA	PA	Rapporteur
PTA			FIPA Request	
RA		ADvISER compliant	FIPA Contract Net	FIPA Request
PA		FIPA Request		
Rapporteur		ADvISER compliant	ADvISER compliant	

Briefly, the ***PTA*** agents receive orders from the MES system and launch ***PA*** agents to execute the production requests, exchanging the product, and process planning information by using the *FIPA Inform*.

The ***PA*** agents interact with the ***RA*** agents during the execution of the process plan, according to the current production line conditions and queries about the progress of the plan execution.

The ***PA*** and ***RA*** interact with the **Rapporteur** agent to provide feedback information about the production execution and to receive optimized scenarios to improve their execution. The presence of a **Rapporteur** agent is optional and aims to provide global decision-making strategies based on data analysis methods that exploit information collected from individual agents.

The ***RAs*** interact between themselves during the physical synchronization of production activities, this is a necessary step to guarantee to have a situation where one or more necessary services are provided. From the RA's perspective, they try to get as many services invocations as possible. For this purpose, they are continuously aware of the competitiveness of their services and the ability to execute a service reconfiguration when an opportunity to improve their services is identified. Accordingly, the RA embed several intelligent algorithms to handle the when and the how phases.

Note that, to drive the production smoothly, no other product can be on the specific station at the same time. The agents need to collaborate in order to find out the best available time to execute such a service. In fact, it may even be that the best solution is to change the services on particular machines to start offering specific services. System performance benefits from a cooperative approach to readjustment of services performed by intelligent software agents. This leads us to the necessity of the next agent type.

The **Rapporteur** uses the collected historical data, and without critical time restrictions, simulates several scenarios to get conclusions about the service reconfiguration proposal's benefits. In the end, the results are sent to the RAs.

Although every agent is assuming control over its scheduling, its knowledge of the world is partial. This partial limitation of environmental knowledge promotes interaction with other agents. For instance, when the agent does not have sufficient abilities to perform some action (e.g., adaptation) the agent interacts with the other agents, this general behaviour of the multi-agent system emerges from the cooperation among its individuals placed throughout the agent production line.

This social behaviour requires the contribution of the local behaviour and the sharing of knowledge among agents. These two aspects together represent an essential requisite to address smart manufacturing. To our particular context, this ability is fundamental to design collaboration for:

- Coordinate the actions of each agent according to the dependency of the product and the production plan.
- Reconfigure, adapt, and optimize production control.

The collaborative scenario also creates a smart way to manufacture a particular product, it concedes the realization of a modular, distributed, adaptive global model, and reconfigurable control platform.

3.6.4 Agent Knowledge Base

A typical distributed system involves several parts, where each part only has a partial view of the system, requiring the need to interact among them to exchange the shared knowledge. The use of ontologies by the ADvISER agents, besides, to guarantee a standard structure of the knowledge exchanged and increase how the knowledge is expressed [141], it also supports reconfiguration without human intervention by using semantic web technologies. Technologies like the semantic web are known to fascinate the attention of researchers in many fields, in particular in the industrial manufacturing domain [7, 96, 164] to express the ontology-based reconfiguration approach. The ADvISER ontology will formalize the semantic description related to:

- The resources that are available in the production line.
- The product and process models that describe how to produce a catalogue of products.
- The description of the production history, including the results from the inspection tests and supporting the traceability process.

Furthermore, the ADvISER ontology allows defining rules and reasoning for the relationships of these concepts.

3.6.5 Agent Architecture Type

Rather than only reactive behaviours, the ADvISER agents follow a design that gives a new dynamic methodology, aggregating the reactive and proactive behaviour. Both behaviours oversee different manufacturing situations.

Dedicated reactivity behavior deals with reactive real-time online control, triggering real-time notifications to address crucial events (e.g., service breakdowns), while proactive behavior pro-motes beneficial service (re)configuration events (e.g., change of service modifications).

Employing the design of a hybrid agent type consists of two types of reconfiguration phases: offline and online mode. These two phases exhibit distinct characteristics, which helps separate the mechanisms to be used:

Offline ↔ Design-time allows to make modifications in offline mode, which is ideal for services that are evolving, and in parallel is proactively planning-ahead identifying and preventing problems.

Online ↔ Dynamic avoids stopping the system, which means that the system itself can be adaptable and run at the same time, allowing ADvISER to respond quickly to the problem.

Based on self-reconfiguration methods, a general method was developed to support the selection strategy in a predictive-reactive manner. The ADvISER reconfiguration control mechanism set in on the agents also considers two types of reconfiguration phases, namely *offline* mode, and *online* mode. These two phases exhibit distinct characteristics, which helps to separate the mechanisms to be used.

While the reactive mechanism is focused on handling uncertain events that require to react quickly and automatically to avoid some type of disturbance. On the other hand, situations in which it is necessary to reconfigure, but the effectiveness of the reconfiguration is preferable to the response time, therefore, in this perspective, the intelligent offline mechanism is preferable. This balance concerning the *"best"* vs. *"rapid"* reaction deserves more detail, so in the following chapter, we will deepen these concepts in the proposed approach.

3.7 Learning

The major industrial transformation we are witnessing has led to the creation of numerous entities distributed on a manufacturing line via IoT or CPS to digitalize devices and processes towards a "Smart Factory" concept. This evolution has led to the production of a considerable amount of data for better and more accurate decision making, but this step introduces new techniques and technologies to collect data and efficiently analyse a large data set in real-time. In manufacturing domain, we are also witnessing a huge growth of AI and ML algorithms across several topics, such

as on the creation of production scheduling, more accurate condition monitoring or preventive maintenance. In these cases, besides performing automatic tasks, ML can assist the operator in identifying abnormal measures and produce warnings to the operator or even to generate activities to execute. Figure 3.13, briefly describes the three main categories of ML algorithms, *i.e.*, supervised learning, unsupervised learning, and reinforcement learning (RL).

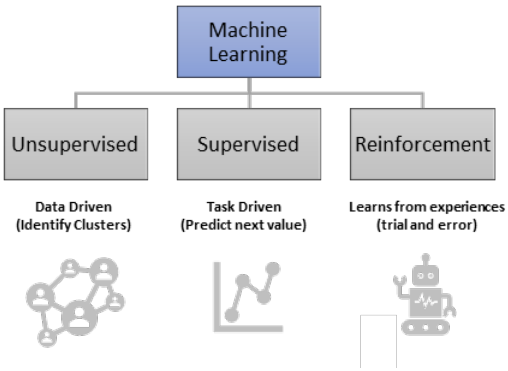

Figure 3.13: Types of Machine Learning.

The unsupervised learning is used in cases where the challenge is to find implicit relationships in a given unlabelled dataset, in some cases, the algorithm is learning as the data is coming in (*i.e.*, as known as *Online*).

While the supervised learning machine algorithms are used to recognize patterns, in this case, the machine is taught by example by the operator with the labelled training set. In the agent technology, this can happen in *Offline* mode, which means that the agent will learn based on a static dataset [10].

By last, the RL algorithms are not like any the other types, we do not have labeled or unlabelled datasets. The RL learns from past experiences and starts to adapt its actions in response to the situation to achieve the best possible result; in other words, it teaches through trial and error. The RL is frequently used in robotics where a robot can learn to avoid collisions by receiving negative feedback after bumping into obstacles.

Machine Learning Algorithms on Industrial Agents

Despite the simplicity and very intuitive trial and error type of behaviour of the RL algorithms, implementing this solution into industrial agents should be wisely selected. Based on the experience gathered from the development and installation of the agent-based solution in a real industrial production line, we were(are) able to learn some regarding developing agents with learning capabilities [112].

The development of solutions based on MAS needs to be validated and verified before going to production. Under this context, if RL algorithms are selected due to its potentialities, we need to have in mind that they are based on trial and error, which means that we are creating space for testing solutions that are not suitable, but necessary to train the system. In order to convince the industry of the reliability of such algorithms, we need to perform offline simulations to ensure the correct behavior of the system. Nevertheless, simulations and tests do not replace the use of online

experimentation in the industrial production line because there are situations that arise only in real industrial environments.

ADvISER Agents and Machine Learning Algorithms
One of the objectives of the ADvISER multi-agent system is its ability to learn over time, thus improving the effectiveness of actions. A fundamental way is after any reconfiguration happens, it is evaluated whether the actions taken were useful to improve or maintain system performance.

Agents may possess many features in various combinations. In ADvISER will be integrated either the learning in *Online* e.g., by data mining when possible, from data which are continually collected through interaction with the environment (*e.g.*, RL) in the dynamic reconfiguration mechanism of the agents, or *Offline* learning refers to training processes (*e.g.,* for pattern recognition) prior to productive agent usage [10]. In this case, the historical analysis will be integrated into the dynamic reconfiguration mechanism of the agents.

3.7.1 Learning on the Machine and in the Intelligent Product

When it comes to learning in intelligent manufacturing, different entities can learn by distinct processes. This situation tends to push the developing of the software agents basically into different types of agents that not only reflect their mental capability and it also impacts the range of services that offer or invoke.

As proposed, reconfiguration in ADvISER architecture lay on top of two main agent types [175], namely: *Product* and *Resource* agents, which have different reconfiguration needs.

3.7.2 Learning on the Agents

The type of intelligence used in the reconfiguration is not only from the perspective of the service provider, that is, the agents that encapsulate the physical operations provided by machines as services, (e.g., drilling or welding operations).

The learning capability follows in the intelligent product from the perspective of the service consumers that need to invoke machine services to execute their products aiming to meet the production demand. The Product Agent focus on changing the structure of a composite service and the RA cover the improvement of the service's behaviour, the changing of the services' catalog and the changing of the structure of a composite service. This changeability in distinct entities creates a distributed system, with the individual agents running the reconfiguration mechanism. Upon this aspect, we can guarantee partially reconfigurations by the decentralization feature and also evolve dynamically by the on-the-fly reconfigurable system.

In general, from the perspective of achieving dynamic reconfiguration, the ADvISER framework's agent network has consistent mechanisms and interactions from autonomous and intelligent agents. Obviously, ADvISER distributed agents should not react to the same or similar situation, always in the same way and in the same amount of time. On the contrary, it seems reasonable to use local knowledge or knowledge of other agents to analyse or explain, each example of training to optimize future performance. Therefore, agent intelligence refers to a specific and conventional set of mechanisms that includes the ability to reason and reuse the experiences learned to ensure performance in reconfiguration decisions.

Learning as a Service

In order to implement such a system, it takes advantage of the various ML algorithms in the AI community, some algorithms being more appropriate than others to support real-time learning design. For example, a RL mechanism is quite effective as it does not require the need to know explicit system models or domain-specific knowledge. Additionally, other techniques like supervised learning algorithms that might be useful for predictive analytics requires a training phase, then an ability to learn historical models to make behaviour models and use these models in runtime to perform anomaly detection on new data generators.

Taking advantage of the architecture developed in ADvISER and taking advantage of the SOA principles, this type of algorithms and techniques can be offered as services.

Regardless of whether it is performed locally (at the resource level) or globally (at the rapporteur level), algorithms and techniques remain accessible as services. In fact, this allows changing the ML algorithms in an easy way offering the possibility that the agent can easily invoke other ML services. This perspective places a great deal of emphasis on services, as is already the case with Machine Learning as a Service (MLaaS) paradigms.

3.8 Summary

This chapter introduces the purpose of the ADvISER architecture, identifying unique opportunities to improve production performance. ADvISER was designed to continually adapt and evolve in the best possible way, accomplishing suitable reconfiguration goals in a complex and dynamic environment. This is an ambitious task for several reasons:

- It involves understanding the mindset behind the reconfiguration. Based on this, a set of essential principles for designing a framework such as an ADvISER is presented.
- It requires comprehending the requirements for designing an effective system of service reconfigurations, such as guidelines for reconfiguration.
- It ensures the essential industrial requirements that are assigned to the reconfiguration cycle for more efficient reconfiguration.
- It builds the multilayer manufacturing framework structure under the assumption that manufacturing devices can depict common interfaces in the form of services.

This chapter provides a first overview of the use of the ADvISER framework for automatic reconfiguration control. With the continuous evolution of the system, some remaining challenges need to be addressed, such as the adaptability of the reconfiguration mechanism throughout the evolution of the system. Several methods will be explored in the next chapter, particularly in multi-agent architecture.

CHAPTER 4

SERVICE RECONFIGURATION MODULE

"everything needs to change, so everything can stay the same"

Tomasi di Lampedusa in 'The Leopard'

In order to make any meaningful reconfiguration, we must know where we are going. Adaptation and improvement are two concepts that are spread across the topics relating to manufacturing and are often used in numerous contexts and situations. However, how exactly do they work and what are the differences among the methods of reconfiguration?

In order to address this question, it is presented a service reconfiguration module that represents a fundamental pillar of ADvISER. Also, it is presented in this chapter the detailed design of the multi-agent architecture involving the built-in reconfiguration mechanism. To answer the previous questions, a service reconfiguration module that represents a fundamental pillar of ADvISER was designed.

4.1 Engineering a Self-Reconfiguration Mechanism

Charles Darwin, in his book *"On The Origin of Species"* in 1859 [38], advanced a theory of evolution and survival of the fittest that can be applied to many domains, including software as well. Many engineering solutions like software architectures were designed to be able to adapt and evolve to new opportunities in order to stay competitive and survive. This is the collective thinking of most of modern enterprises and has been inspired by nature. To make an analogy, from the perspective of the manufacturing companies, those with the production control software able to change or evolve are those that better respond to volatile environments.

New paradigms and ideas, such as the architecture proposed here, are part of a necessary effort to support a faster and more autonomous reconfiguration of reactive production systems. Although there is evidence that agent-based technology enhances agile manufacturing operations based on flexible scheduling, they do not guarantee optimal production schedules like conventional approaches. This issue is one of the problems why industrial companies are not as welcoming to agent-based technologies, due in part to the lack of industrial experience [53]. Seeking to overcome this challenge, a hybrid agent-based optimization approach has been devel-

oped, combining the benefits of optimized conventional scheduling approaches with agent-based element reactivity. It hard to imagine a self-reconfigurable system that manages the manufacturing operations effectively without thinking in a reactive and proactive manner. The responsibilities of dynamic reactions and proactive planning of the RA *3.6.2.4* allow to design reconfigurations to adapt a running system if possible, in real-time. The real-time system works on the premise that the deadline of each task is met [20]. A current practices to avoid violate this rule consists in building systems that consider local scheduling vs global scheduling working in on-line or offline mode [23]. Unlike the online that runs whenever a task is requested, the offline allows to sets the plan before the execution time, using information specifying the future behavior of the task [23]. In this sense, the development of the reconfiguration considers two basic strategies [167]:

- The first phase considers generating reconfigurations in offline mode (*i.e.*, design-time) based on the current manufacturing state, leveraging from the optimisation mechanism in the simulation's tools to explore scenarios (**IR 3.2**).
- The second phase is responsible for the monitoring of the manufacturing system and the inspection of the need for adaptation in an online mode (*i.e.*, runtime) (**IR 3.1**) .

Service Reconfiguration Module meets Industrial Requirements

Regardless of the phase to reconfigure (*i.e.*, design-time, runtime), the ability to reconfigure must include requirements that to some extent fit the concept of a reconfigurable cyber-physical component. All proposed requirements are essential, but for the service reconfiguration module we focus on the following:

- **Self-adaptation & Responsiveness** - because the module must employ adaptation behaviour, self-adaptive methods (**IR 4**) are considered for adapting the system behaviour of each agent. Also, because we need responsiveness in this module, the (**IR 3**) that joins reactivity and proactive characteristics are considered to manage the operational concerns.
- **Learning** – is another essential requirement for the use of the service reconfiguration (**IR 5**). The objective is to improve over multiple runs, based on the learning of the past-experiences and of offering accurate reconfigurations. As expected, this permits to meet better flexibility and reconfigurability in runtime.
- **Autonomy** – is a crucial requirement to consider (**IR2**). The main point comprehends the decision initiative as a proponent of the change with minimal assistance of external entities. As previously designed the role of the ADvSIER agent consists in ensuring correct execution of the system, thus since one of the objectives is to deal with external services, if these are not working correctly, the agent must guarantee that the system can continue running in the absence of these services. This means that the agent can orchestrate the decision and select not to consider using the offline mode. The ability to operate in the on-line mode in the absence of the offline is guaranteed.

In brief, the core of the service reconfiguration employs several IRs described previously. Based on these requirements we can develop in detail the distinct phases of the service reconfiguration module.

4.2 Offline Service Reconfiguration

As described in EFFRA report on the research and innovation strategies for the factories of the future [55], considers that "*What-if*" algorithm analysis is indispensable. This type of algorithms will help design, in offline mode, configurations that will potentially drive the system into stable and better states [143, 154].

Having this in mind, the *offline* service reconfiguration is going to benefit from the What-if simulations principles developed with multi-agent system, as it was performed by the author [109].

4.2.1 Reconfiguration Planning

The production planning refers to the elaboration of a plan, using an algorithm to optimize a problem subject to certain constraints according to a set of criteria, which contains decisions about the products and their quantities to be produced in the planning periods and about the needed capacity of the production environment, (*i.e.*, number of production lines, machines and services).

The objective is to maximize profit considering the possible modifications of the manufacturing supply. Typically, these problems are hard to be solved, requiring significant computational resources and time (*i.e.*, NP-hard problems), which leads practitioners to use mature and robust computational applications (*i.e.*, mathematical optimization solvers) that may implement different optimization algorithms, ranging from linear programming to meta-heuristics, to solve a complex problem determining the optimal solution for given constraints. However, despite the achieved high optimization levels, these solvers lack the responsiveness to achieve solutions in short-term and to produce dynamically different solutions by varying the problem's constraints. Many algorithms, such as, Ant Colony Optimization (ACO) or Genetic Algorithm (GA), are also being used aiming to achieve solutions in a shorter time.

Agent-based solutions have been reported oftenly [9, 15] to implement the optimization algorithm to provide a way to achieve an optimized solution in a more robust, flexible and agile manner. In spite of the introduced responsiveness, this type of strategy (multi-agent system or heuristics) might miss the achievement of an optimal solution. In this way, the multi-agent principles can be integrated with mathematical optimization solvers, combining the maturity, robustness, and optimization of the solver with the flexibility and responsiveness of the agent-based solutions. An example of the use of this hybrid approach to building optimization production tools is illustrated in [157].

What-if strategies were already used in other domains to offer the decision-makers with valuable information to support the decision-making (*e.g.*, a what-if simulation applied for an efficient failure-based maintenance decision support, by merely varying the inputs and foreseeing how the system behaves [18]) as illustrated in Fig. 4.1.

Even though this type of approach is not traditionally used in the production planning, the offline service reconfiguration module can benefit, generating production planning scenarios and simulating the alternatives solutions to send to a pool of candidates configurations that will be used by the RA agent after providing them for the user's evaluation if necessary. RA manages continuously to provide the most promising reconfiguration planning solutions.

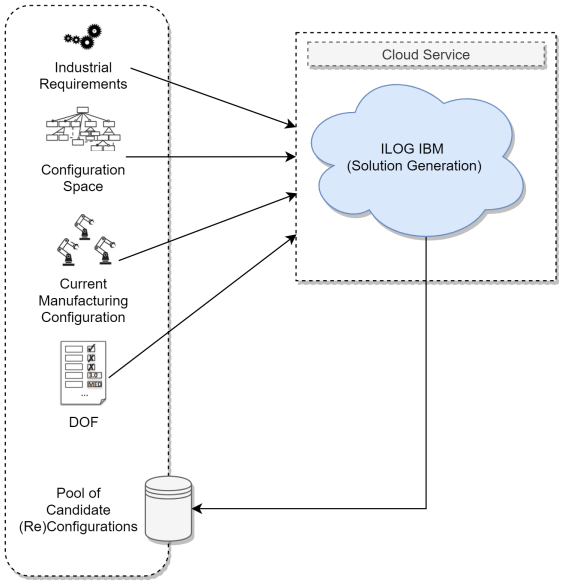

Figure 4.1: Overview of the what-if simulation in *offline* mode, connected to the agent's knowledge base.

As a small example, [109] contains a detailed strategic planning system that combines the MAS principles and mathematical optimization solvers, aiming to provide mitigation strategies for the ramp-up production phase of sophisticated and highly customized products. This idea, developed in the ARUM project (Adaptive Production Management - http://arum-project.eu), was validated by considering a real case study related to a manufacturing company that produces modular equipment used during the airplanes' flights.

4.2.2 Playground for The What-If Simulation

The strategic reconfiguration planner provides the production planning functionality related to the long-term plans. Since there are no time constraints, more information can be given to the solver, the demand, for instance, might be given, or estimated, for a specified planning horizon. In brief, the proposed approach is composed of a set of steps, each possessing individual functionalities to the accomplishment of the reconfiguration solutions:

1. **Definition of the planning requirements:** this stage specifies the information about the system and stipulates the Degrees of Freedom (DoF);
2. **The generator of planning scenarios:** generates scenarios for production planning, exploring different DoF costs/revenues;
3. **Planner:** apply optimization algorithms to find a solution for the reconfiguration planning problem of a certain scenario;
4. **Simulation mechanism:** responsible for assessing the production plans through simulation to anticipate the stochastic behavior of the production system.

The behaviour of the offline service reconfiguration emerges from the What-if simulation interaction, as shown in Fig. 4.3.

The what-if simulation interaction works as follow: (*i*) after considering the problem description in terms of the mathematical model and the DoF that will allow exploring different reconfiguration

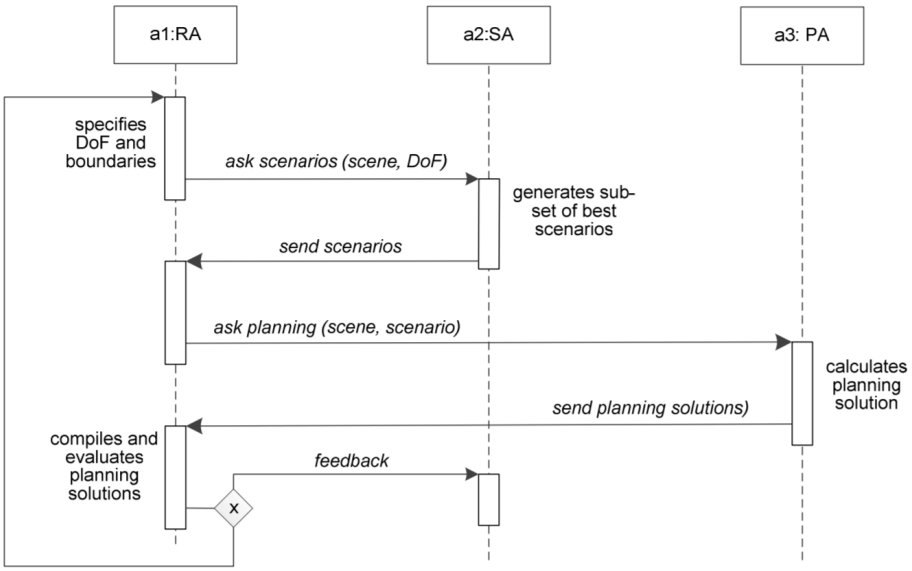

Figure 4.2: Interaction pattern for the What-if simulation.

scenarios the initiator requests for scenarios to the generator; (*ii*) the generator produces a set of scenarios considering the boundaries of the DoF pre-established (such as the number of machines, production lines, services) and sends it to the initiator; (*iii*) the initiator after receives the list of scenarios requests to be solved to the planner under a specific scene; (*iv*) the planning works based on a solver, (*e.g.*, IBM ILOG CPLEX Optimizer, LP_SOLVER), running optimization methods, (*e.g.*, Mixed Integer Programming), considering the specific problem formulation and the defined scenario; (*v*) when the initiation receive the planning solutions the next process consist in compiling and evaluating, saving them into a pool of possible solutions to be available to the agent execute.

4.2.3 Generate the What-if Scenarios and to Analyze the Planning Solutions

One of the primary responsibilities of the scenario generator is to generate a set of scenarios to be used by the agent. The scenario manipulates the range of the DoF aiming to select a subset of scenarios from the search space (*i.e.*, all DoF possible combinations) that better represents the production planning solutions to be tested (aiming to reduce the response time). The idea is to select only the most promising scenarios avoiding the need to test all possible scenarios (as illustrated in Fig. 4.3).

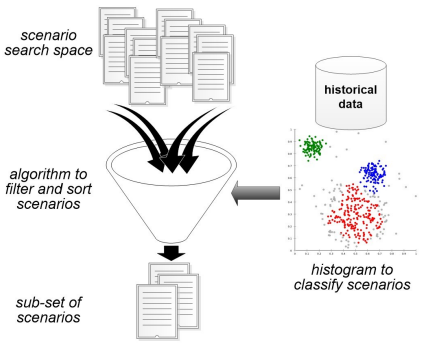

Figure 4.3: Scenario generation process.

The decisional engine behind the scenario will generate scenarios based on the DoF (and particularly their boundaries), and considering historical data from previous iterations, where similar inputs can produce similar outputs, avoiding the testing of weak or already non-valid scenarios. It is important to narrow the search space to be covered by the strategic planning, by only considering the scenarios with a high probability of leading to the right solutions. Figure 4.3 illustrates the process to generate scenarios, particularly their sorting and selection, based on a kind of histogram learning.

In this process, the scenario uses a matrix for each unexpected problem type (*e.g.*, increase the services), which contains the learning values for each scenario. Since each scenario is defined by the configuration of the different DoF, the matrix has as many dimensions as the number of DoF considered. The range of each dimension corresponds to the boundaries of the DoF of such dimension.

$$\begin{bmatrix} A_{11} & \cdots & A_{1n} \\ \vdots & \ddots & \vdots \\ A_{m1} & \cdots & A_{mn} \end{bmatrix}$$

The learning values are reinforced based on the feedback of the quality of the planning solutions according to the selected KPIs:

$$A_{ij_{k+1}} = A_{ij_k} + f_p - f_w$$

where is the new learning value, is the previous value, f_p is the positive feedback and f_w is the negative feedback. Note that the feedback values (from the human operator or ADvISER system) are dependent on the quality of the planning solutions and can impact the reinforcement process according to how good or bad the solution is. By this, learning can be divided into two distinct phases, where firstly the system is ranking the planned solutions based on the pre-defined KPIs, (*e.g.*, solution cost, reconfiguration time), and secondly in the user's solution perception. Both learning mechanisms are using reinforcement technique as mentioned above but with different reinforcement weights. Proper selection of the reinforcement weights must be performed, but the user's natural solution perception has a higher weight. These weights' choice is based on the fact that strategic planning is not a straightforward process, where the most profitable solution might not be executable. Note for instance a solution that considers an intermittent usage of production lines that is highly ranked by the MAS system. Since this solution may raise social instability, the decision-maker might prefer to discard it over other alternative solutions.

Aiming to improve the accuracy of the learning values about the scenario search space, being better prepared when a request for generating scenarios arrives, the scenario generator also performs exploratory testing in its idle time (*i.e.*, in the background). In this case, scenarios not well ranked or not yet considered in the past should be tested, updating its ranking and avoiding the non-detection of possible right future DoF combinations. When a request to generate scenarios comes from the initiator, and in order to select the most likely scenarios, the scenario generator will analyze its knowledge related to the previous scenarios' results. For this purpose, the generator selects the proper matrix and extracts a submatrix, considering the pre-defined DoF boundaries. Using this new matrix, the scenario generator will, based on the proper learning matrix, extract the n most ranked scenarios for this planning problem type (sorted in terms of the received feedback

reflecting the quality of previous planning solutions) and elaborates the sub-set of more promising scenarios to be sent to the resource agent.

The described approach runs iteratively when an existing problem type already exists in the agent knowledge base. However, when a new problem type appears, the matrix is built from scratch, considering all possible scenarios. Using learning by analogy techniques, it is possible to accelerate the learning phase by considering similar existing problems. Another option is to consider the exploratory mode previously described.

Evaluating and Sorting Planning Solutions

After the generation of a set of scenarios, these are used to perform the production planning solutions by calling the mathematical solver methods of the planning class. Each *planning* solution (associated with one scenario) is evaluated and ranked according to a KPI, being offered to the decision-maker a summary of the best solutions, based on two stages.

The initial step is related to removing the planning solutions that do not fulfil the initial requirements, (*e.g.*, solution cost). Note that initially, besides the DoF boundaries, the decision-maker also needs to define the number of planning solutions, n, and the minimum acceptable values for each KPI, *i.e.*,

$$KPI_i < \delta_i$$

The second step is related to sort the achieved planning solutions and select the n best solutions considering each KPI individually and considering a multi-criteria function that weights several KPIs. The n solutions for each KPI and multi-criteria function can be shown to the decision-maker. The weights can be defined by the human operator at the beginning of the procedure and benefit from automatic decision from that point on or be shown to the decision-maker using a spider diagram, with the score of each solution for the KPI.

Finally, the automatic or through the human operator selection the reconfiguration solutions are select or, if not satisfied, discard some provided solutions and gets again, in an iterative manner, another set of n best solutions. Additionally, the decision-maker can take a more disruptive decision and adjust the DoF boundaries and start a new iteration in the what-if simulation. This iterative procedure can be performed manually by the decision-maker or automatically by the agent-based system, allowing to adjust the DoF boundaries for each round.

In the end, the planning agent sends the achieved proposals to the agent. The what-if simulation functionality supports the decision-maker to analyse and simulate what happens if some DoF are changed, and consequently be more prepared to select the best strategy to mitigate the unexpected event.

This idea was tested within an EU project [111]. The objective is to illustrate the selection of the KBFs by the production manager to perform a What-if game analysis by simulating the change in KBFs and assessing in real-time the impacts of the KPIs for a given station. As illustrated in Fig. 4.4, the user can play with several DoFs related to KBFs located at the bottom, namely the setup time and performance boundaries, sliding the DoF's bars.

Figure 4.4: Selection of DoF and visualization of KPIs during the What-if scenario [111].

The KPIs describing both the overall system performance and the specific workstation behaviour are plotted on the top. The tool calculates the alternatives for the specified set of defined KBFs, and shows them in a spider diagram, each one presenting the real-time impacts of the business factors in the operational indicators. The decision-maker can play with the KBFs values and change them using the sliding bars to verify their impact on the operational production indicators.

Summarizing, the assessment performed during the operation of the KPI monitoring tool in the real industrial factory plant allowed to validate its correctness and verify the benefits of the tool, particularly to assess the increased customization and reconfiguration opportunities

4.3 Stages of the Online Service Reconfiguration Module

All agents distributed around the environment end up being important because of their autonomy, no matter how small their decisions, therefore the core of the reconfiguration mechanism is like every single agent and based on specific modules, which are highlighted in Fig. 4.5.

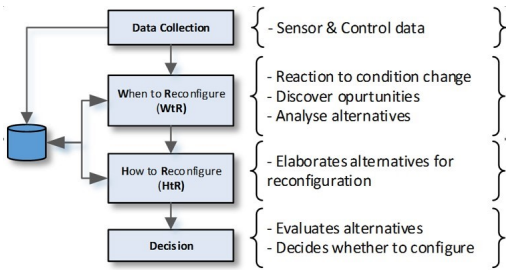

Figure 4.5: Overview of the service reconfiguration modules connected to the agent's knowledge base.

As shown in Fig. 4.5, and as already described in Chapter 3, the agents are continuously monitoring the system in a continuous manner, which gives us a feeling of continuous feedback that is essential for an online service reconfiguration system, which consists in the following modules:

- The *"Data Collection"* module performs real-time monitoring of multiple available sources, *i.e.*, specific services and properties of the services.
- The *"WtR"* module receives analyses, in real-time, the collected data related to the service performance, and is responsible for identifying and predicting the need or an opportunity to perform a service reconfiguration.
- The *"HtR"* module has the responsibility to identify the steps either to register for a new configuration, or through evolution or adaptation.
- Finally, the *"Decision"* module is responsible for deciding about the implementation of the best service reconfiguration alternatives.

Looking up close, each autonomous agent follows a specific behaviour of autonomous computing, which is a local loop to building continuous reconfiguration strategies.

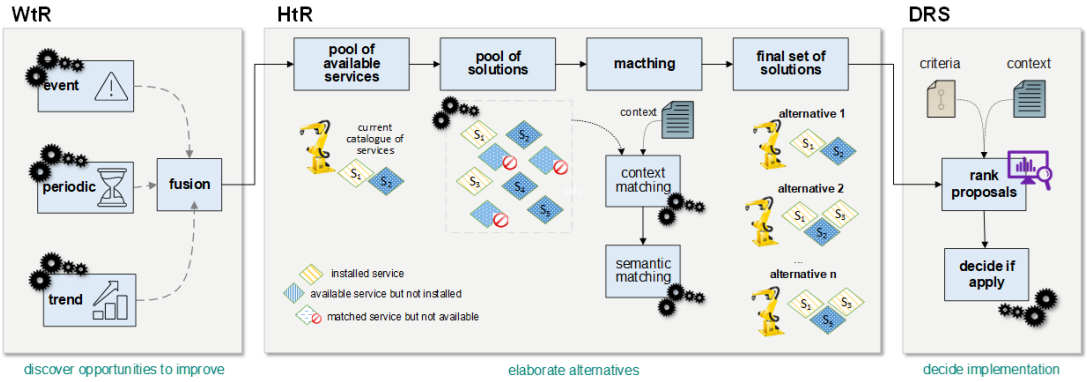

Figure 4.6: Service reconfiguration modules built-in in each agent.

As illustrated in Fig. 4.6 the service reconfiguration module comprises the remaining modules, *i.e.*, WtR, HtR and DRS phases [170]. The different phases will be detailed in the next sub-sections.

4.3.1 Data Monitoring

Data can empower decisions, so the necessary methods to capture the data to support the decision are implemented in ADvISER vision. During the runtime data is collected by the agents from the services to be analyzed under a specific context. The result can be directly displayed to the operator through a graphical dashboard for validation of possible reconfigurations, to perceive anomalous behavior in the real-time monitoring and even, to support the most inexperienced operators by illustrating how to perform the actions.

Machine intelligence

Aligned with Industry 4.0 principles, ADvISER has the capacity to integrate the human-in-loop but considering that there are quite complex scenarios that are even humanly impossible to solve it is believed that it is advantageous on various levels to supply the manufacturing resources with machine intelligence, especially with self-adaptable (*i.e.*, **IR4**) permitting a better regulation of the manufacturing operations.

Moreover, it is helpful for the agents to work with the services in an intra-layer and cross-layer perspective to facilitate the changeability, the system can be more productive, while operators might receive higher-value tasks that only human beings can perform.

Reconfiguration Context

One of the main characteristics of a decision is the context in which it occurs, such as device context, current production, machine performance. An important factor is the context of the time, as in dynamic and evolutionary environments it may theoretically never be possible to make two decisions with similar contexts. However, it is well known that changing decisions are crucial for a system and should, therefore, be considered under the context in which they were made. From this perspective, the goal is to capture changes in the operational environment and offer all the variables necessary for ADvISER to make the best possible decision. In some situations, resource-type agents may indicate changing conditions with simple basic rules (such as new services or agents in the environment), yet the concept is based on monitoring all of the machine services' health, indicating certain situations that can be analyzed in the next module.

4.3.2 When to Reconfigure Phase

Once the data has been captured, the next step is to look at the data and discover opportunities to reconfigure. This specific process aims to design **ADvISER Feature 6**. Looking at 4.7, this phase represents the first module of the service reconfiguration.

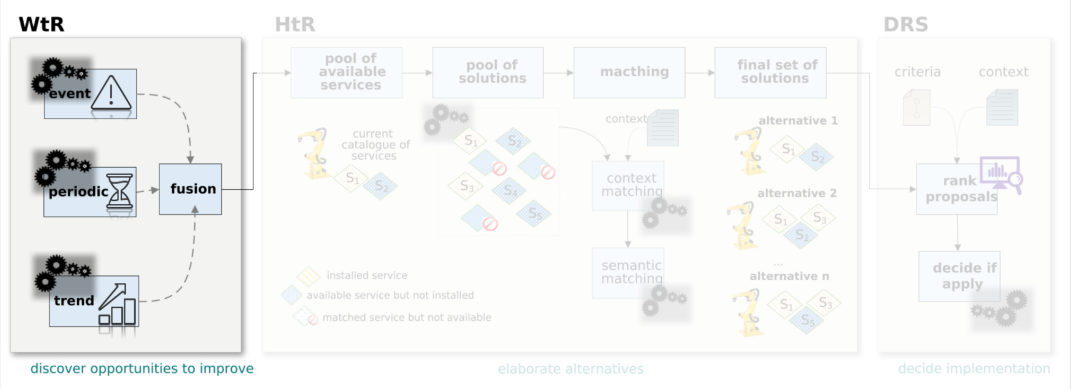

Figure 4.7: Focus on the WtR phase of the service reconfiguration module.

The agent's focus moves continually across many phases, particularly in this phase, the agent is observing and analysing several policies.

4.3.2.1 Requirements to Capture Changes in the Operational Environment

Before an overview of strategies, it worth explaining some fundamental aspects of activation mechanisms, for instance, it is recommended to consider the **clarity** of the data (as already mentioned), the **applicability** and the **implication** aspect [49].

- **Applicability** – the system must be aware of the applicability to avoid the activation of useless reconfigurations, particularly in domains like manufacturing, where a reconfiguration decision has a more significant impact, E.g., some works focus their research on trying to keep the system with the minimum number of adaptations to increase the life of components.

- **Implication** - it is indispensable to know the **consequences** of each activation in the future, and their implications: what are the implications of choosing a bad strategy? One obvious answer is the benefit of adaptation, but this is not just the problem of a bad decision. A wrong decision can lead to monetary losses because the desired state has not been reached or it damages the system, or even unnecessary human effort is required in interventions or predictive models with little confidence due to various adaptations.

Autonomous systems need to select specific strategies, optimizing the number of times they adapt, always trying to realize their implications. Proactivity in its essence produces alternatives that can be consulted over and over again, which is not beneficial to be constantly reconfiguring. Mastering proactivity and learning mechanisms (IR5) can thus help reduce the number of reconfigurations for those where the system knows it is beneficial or necessary. Knowing the type of applicability in which they can be used and how to use it will reduce the future implications. This requirement is in line with the following IR (**IR3**, **IR4**, and **IR5**).

Finally, a relevant aspect of the "When to reconfigure" (WtR) phase concerns the reason for the reconfiguration. As it was described, the system reconfigures based on two reasons: to adapt or to evolve, but these are too generic. Thus, the next chapter describes in depth such motives.

Understanding Why to Reconfigure

Reconfiguration is carried out by adaption or evolution processes. The responsibilities of both possibilities require the understanding of the reasons and steps to adapt, which serves as a bridge to connect the problem to be solved with the most sensible resolution of the moment. As a result, to achieve good efficiency in the selection of the strategy, it is required to have a clear notion of the necessary steps and tasks to adapt from the current configuration "*as-is*" to the desired service configuration "*to-be*" without worrying about the technical details.

Within this context, the agents employ adaptation self-* properties to increase the systems' flexibility, responsiveness, reconfigurability, and autonomy (see **Feature 5** in Chapter 3). These mechanisms adopted by the agents allow dealing with the reconfiguration issues in a continuous form. Technically, this situation of selecting several functionalities of the self-* (-configuration, -healing, -optimizing, -protecting) might become chaotic and complex to handle if all are executed at the same time. In some cases, the simultaneous execution might not always be advantageous, since there are mechanisms that affect others with their activation, for example, performing self-healing is going to have an impact on self-optimizing. Under this perspective in ADvISER implementation, it was necessary to be aware of the use of the mechanisms, particularly when they are activated.

With this in mind, the final choice will be left to the agent in the reconfiguration module, which will consider the reasons and steps to reconfigure in runtime. The WtR phase belongs to the reconfiguration module, which verifies what happens after the continuous monitoring and analysis of the collected data. The next step is to understand if there exists an opportunity to reconfigure. This is carried out by the activation mechanism, using different triggering strategies, which will be detailed below. Generally, once the reconfiguration module receives the data, the WtR combines the self-* functionalities (*i.e.*, **Feature 5**) with the triggering strategies (*i.e.*, **Feature 6**) into one single process to activate the evolution.

4.3.2.2 Triggering Strategies

The reconfiguration analysis on the WtR model depends on three different triggering strategies for the online detection of possible situations to reconfigure [170], namely, event, periodic and trend. As illustrated in Fig. 4.8, in any of these cases, the WtR module receives data from the Data Collection module or retrieve it directly from the database.

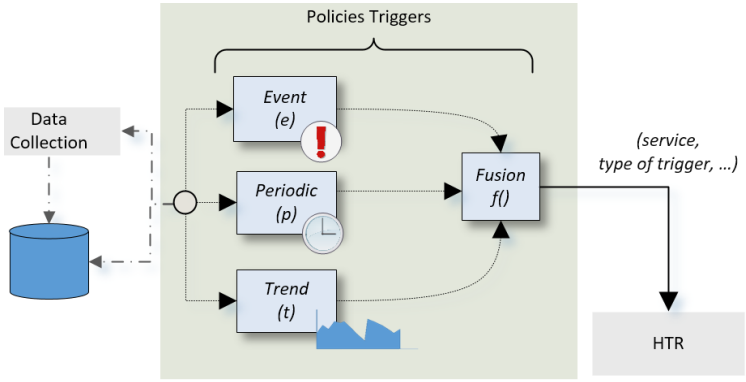

Figure 4.8: Internal architecture of the When to Reconfigure module.

- **Event:** uses an event-driven approach to detect the occurrence of events related to the system's condition changes, *e.g.*, a resource failure or the removal of an existing resource/service. This strategy allows a good reaction facing unexpected events.
- **Periodic**: uses a periodic check to verify the current service performance and decide about the opportunity to reconfigure. The triggering time interval should be dynamically adjusted to fit the system dynamics better, *i.e.*, by increasing or decreasing this value, taking into consideration the application of proper machine learning algorithms. As an example, Q-learning [210] is a suitable method to implement this feature since it provides a positive/negative reinforcement feedback that allows converging to an optimal value at each moment.
- **Trend:** uses machine learning methods to detect, as early as possible, a trend or pattern in the service performance, allowing the prediction of an eventual problem, and consequently to identify an opportunity to implement a service reconfiguration procedure earlier to mitigate this possible loss of performance. Anomaly detection, cluster analysis, and structural break are examples of algorithms that can be used in this strategy.

Triggers indicating opportunities to proceed with reconfiguration, generated from different strategies, can happen at the same time concerning different features and contexts, without any explicit interference between them. When this happens, the fusion module is responsible for joining the concurrent triggers, selecting the most critical one according to a set of trigger rules and firing one event to the HtR module that will handle the elaboration of the space of service reconfiguration alternatives.

Each trigger policy applies a different strategy generating different triggers over time. Some triggers pointing to reconfiguration may happen at the same time regarding different features, without any explicit interference between them. When this happens, the fusion module is responsible for merging the generated triggers and redirecting the resultant one to the HtR module. The behaviour of each one of the trigger strategies can be dynamically adapted, by using appropriate learning algorithms, to optimize the accuracy of the module, *e.g.*, using the feedback about the reconfiguration suggestions (collected by the Data Collection module) to adjust the previous

threshold parameters' values. Next sub-sections will detail the three triggering strategies.

Event Triggering

This strategy is activated by using an event-driven approach that detects events related to the system's condition changes. What is significant for this situation are two groups:

- **Internal events** – depending on the behaviors (*e.g.*, self-healing), the agent may purposely trigger internal events to regulate its services, for instance, the removal or addition of services. There is also another type of internal event, but contrary to the first one, this happens involuntarily. For example, a service that failed is identified by the self-monitoring process of the agent that detects this unintentional situation and fires and the internal event.

A major characteristic of this trigger is the ability for a first reaction by the agent to facing critical events **IR 3.1: Reactive**, in analogy with the corrective maintenance benefits. Basically, since the trigger follows a straightforward strategy, that is, simple and fast, it can respond dynamically and quickly to the disturbance.

The design of external events is the second group that triggers a reaction in the system.

- **External events** – External activations are those more straightforward to perceive, performed by external entities, who purposely want to change the behavior of the system such as, new production order form MES, ERP or adding or removal of physical equipment (*e.g.*, plug-and-produce like).

This kind of event, like the previous one, highlights the changes in system dynamics. If any of these occurs, a reactive event is fired and sent to the HtR module.

The key point is the ability to deal with unexpected data that occurs in the system, which makes it an essential feature in dynamic and unpredictable environments.

Periodic Triggering

In this strategy, the feedback allows to adapt the verification periodicity, in opposition to the event triggering, this strategy occurs on a scheduled basis, being very similar to the preventive maintenance. The strategy performs a periodic check to verify the current service performance and decide about the opportunity to reconfigure. In detail, the algorithm consists of a time interval Δ, which may be dynamically adapted.

A typical non-manufacturing example may be to change the car's oil, where the frequency of maintenance can be calculated according to statistical information about the components and the running process to schedule, *e.g.*, the Mean Time Between Failures (MTBF). Figure 4.9, illustrates how can the MTBF be measured, and how its value is evolving along the time.

MTBF, it is the total working time divided by the number of failures, formally:

$$MTBF = \sum \frac{d_i - u_i}{n} \qquad \text{, for all } i = 1, \text{ for all } n \text{ observations.}$$

Calculating actual MTBF requires a set of observations, each observation is:

- n = Number of observations.
- u_i = Uptime moment, the moment at which a machine began operating (initially or after a repair).
- d_i = This is the ith = Downtime moment: the moment at which a machine failed after operating since the previous uptime-moment.
- So, each Time between Failure (TBF) is the difference between one Uptime moment observation and the subsequent Downtime moment.

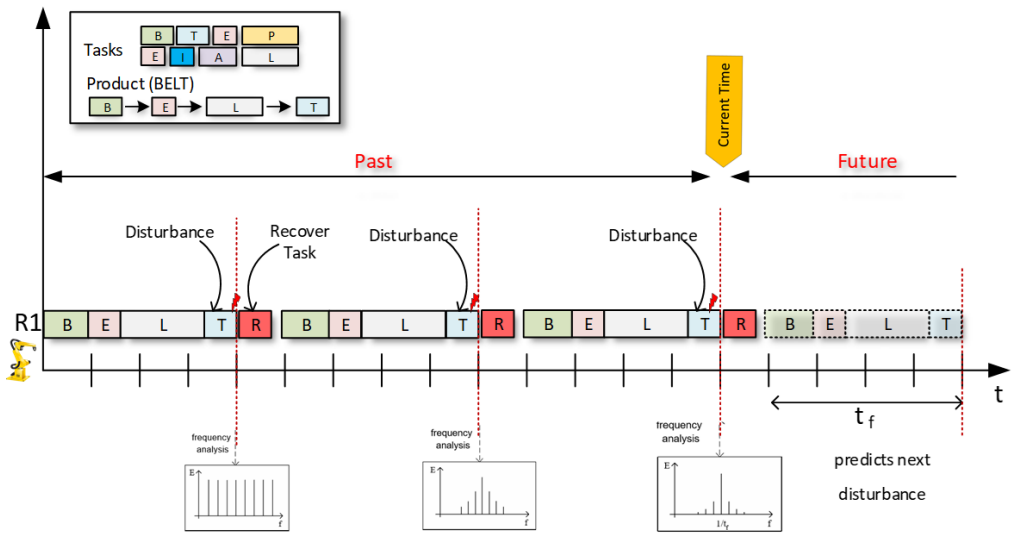

Figure 4.9: Illustrations of a series of disturbance events.

However, this approach only detects failures inside a certain time-frequency. If we change context the time-frequency changes, let us consider for example that the configuration of the production needs to be reconfigured, which leads to different tasks, with different operation times to be produced and therefore the MTBF will become challenging to predict. In order to cope with the lack of accuracy to detect potential disturbances, several techniques are proposed, without forgetting the simplicity of the MTBF measure.

Learning the Periodic Triggering with Q-Learning

Q-Learning [209] is a Reinforcement Learning (RL) technique that learns a state-action function describing the utility of taking action at a certain state (s). The action (a) chosen is the one that maximizes the function Q(s,a), Q being the Quality of a state-action combination. In general, this strategy enables one to periodically trigger an action to perform preventive service reconfiguration, reducing the probability of service failure or performance degradation. The triggering time interval (Δ) should be dynamically adjusted to fit the system dynamics better, *i.e.*, increasing or decreasing, taking into consideration the application of proper machine learning algorithms. In this context, Q-learning is a suitable approach to address this challenge, since it provides a positive/negative reinforcement feedback that handles the system's dynamics, allowing to converge to an optimal value of Δ by mapping the values of each agent in a table state-action, as follows:

The state represents the agent's current knowledge situation. Specifically, in our case, the State is formally defined by:

$$State =< St, \beta >$$

where,

St is composed by the set of services;
$St = \{s1, \ldots, sn\}$, at the instant t;
$\beta =< \xi, \rho >$ corresponds to the description of the production batch;
ξ represents the number of orders;
and ρ the product type.

Agents will take some predefined actions regarding the dynamic update of the Δ value, which will change according to the following set of actions:

$$Action = \langle \uparrow \Delta, = \Delta, \downarrow \Delta \rangle$$

where,

- Increase the time interval (↑ Δ);
- Maintain the time interval (= Δ);
- Decrease the time interval (↓ Δ);

The reward (R) is calculated by considering the action taken in a specific state and the reconfiguration triggering feedback. If the reconfiguration activity had led to a better configuration structure, the Q-value of the state-action pair chosen would be increased. Otherwise, it will suffer a penalty for the chosen Δ value, given by equation (4.1), where n is the reward constant:

$$R(State, Action) = \begin{cases} +n, \text{ if reconfigure} \\ -\left(\frac{n}{2} * penalty\right), \text{ if not reconfigure} \end{cases} \tag{4.1}$$

In general, after a few numbers of interactions, the agent can select an optimal triggering frequency, in a given State in a given context.

Trend Triggering

The periodic approach leads to a reduction of the event triggering strategy since the agent is executing service reconfiguration more efficiently in a preventive manner. Despite the advantages of this strategy, some opportunities to reconfigure due to a degradation of the service performance will be lost, since this strategy is driven utilizing periods and not by the constant monitoring of the activities of the services themselves. Such a challenge is covered by the trend triggering strategy.

This strategy **aims to** recognize a tendency or pattern in the degradation of service performance, anticipating actions to improve its performance or to reconfigure by another more useful. According to some regression models, it is possible to correlate the quality value parameter with the failure occurrence. For this purpose, rule-based approaches and anomaly detection techniques can be used.

4.3.2.3 Decision Rules

The main goal of any rule-based decision approach is to monitor the process performance through the control chart and identify out-of-control conditions. Of course, the specification of a set of rules

is a helpful manner of control if the behaviour of the process is out-of-control. The implementation of the rules consists of applying these rules to a chart control. This simple method involves a graph that comprises the process data plotted in time order, as illustrated in Fig. 4.10.

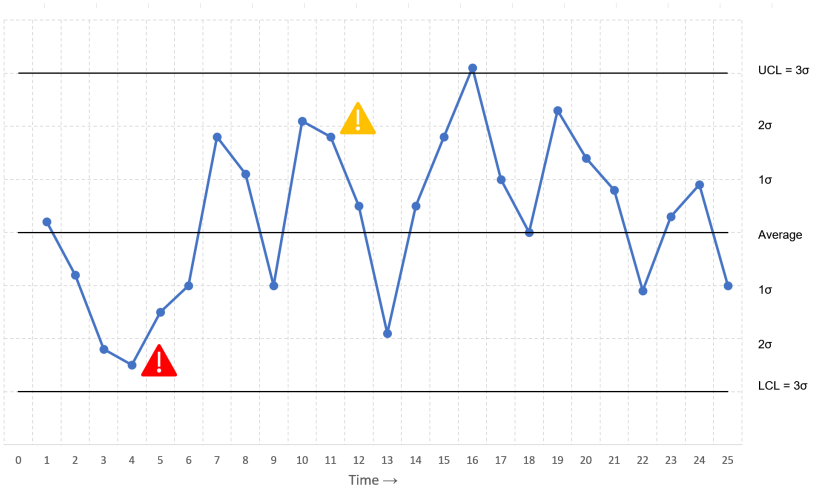

Figure 4.10: Control chart example.

The data plotted on this chart together with the rules represents a typical example of a decision rules technique, where it is relatively easy to identify out-of-control situations like anomalous events and others particular causes patterns, take for example specific rules to detected points beyond the control limits (see Fig. 4.10). A set of limits is specified when the system is performing correctly, such as: upper control limit (UCL), lower control limit (LCL) and average (centre line).

Part of the control charts, of course, is to help to monitor and identify anomalies beyond the limits. An appropriate reconfiguration procedure comes from focusing on the right moment to change, therefore identifying anomalies must be done continuously covering a wide range of potential measures to guarantee a continuous improvement on the reconfiguration process. Thus, the set of criteria should be extended. Other rules for detecting patterns that are not obvious are necessary, such as small variations that may represent instability. So, to overcome this situation, more rules are created, after specific "zones" have been defined in the control chart (see Fig. 4.10). These zones permit to explore diverse situations, like the different types of variations, some of these rules measure points inside specific zones while other rules aim to detect trending behaviors for instance that go across different zones.

The system continuously searches for "out-of-control points", which are identified by the rules that lead to initiating alerts and subsequent activities. A team of experts, which are aware of the process (cause-effect) have already explored possible causes, a plan of actions to eliminate or reduce the effects but more important an idea of the boundaries and rules. Ideally, this analysis should occur at the beginning. Otherwise, after the occurrence of a problem, it becomes difficult to go back and see what has occurred and investigate the cause. Thus, in this context cause-effect diagrams are helpful because they require a specification of the domain processes to understand what needs to be measured and how often it will be measured. The next step is to choose the most appropriate control chart that best covers all possible anomalies. Regardless of the technique, the many control charts are very similar, in fact the rules are the same, the only differences are the number of examples that are required (see Table 4.1).

Table 4.1: Comparison of control chart rules.

#	Control Chart Rule	Western Electric	Nelson Rules
1	n points above UCL or below LCL	1	1
2	**Zone A:** n of $n+1$ points above/below 2 sigma	2	2
3	**Zone B:** n of $n+1$ points above/below 1 sigma		4
4	n points in a row above/below center line	8	9
5	Trends of n points in a row increasing or decreasing	7	6
6	**Zone C:** n points in a row inside Zone C (hugging)		15
7	n points in a row alternating up and down		14
8	**Zone C:** n points in a row outside Zone C		8
9	**Zone B:** n points above/ below 1 sigma; 2 points one above, one below 2 sigma	4	

There are many techniques, Table 4.1 exemplifies two cases for the same objective stability analysis, these are the two well-known decision rules in manufacturing, *i.e.*, Western Electric Rules and the Nelson Rules. Both approaches employ statistical methods to monitor and control processes, the Western Electric Rules is one of the oldest control chart rules, the rest are following similar principles, *e.g.*, Nelson Rules. The benefits of control charts in the industry are clear. However, there are some underlying assumptions to be aware of when adopting this technique that may not be advantageous? Take for instance the following aspects:

- *False Alarms* - bad assumptions on designing the rules (*i.e.*, about which processes to monitor) or rules that are not properly correlated (*i.e.*, some rules may be activated due to inference from other rules, *e.g.*, services shut down because of a rule will possibly affect other rules that are monitoring the performance of a machine);
- *Incorrect control limits* defined at offline mode may produce false triggers at online monitoring.

Any of these situations could lead to a misuse of the "control charts" causing downtime and unnecessary delays. Regardless of the growing attention in this type of control, the tools used do not rely on clear specifications on which are the basic processes and boundaries involved as well as their direct relation. Thus human validation of these limits should be considered to avoid problems.

This idea was tested within an EU project [111]. As shown in Fig. 4.11, the production manager can visualize the main KPIs of the process in the shop-floor (*i.e.*, micro-oven production). Each production stations of the line is displaying the distinct KPIs for each one of them. The target and the actual KPIs values are shown, as well as their evolution trend (positive or negative). A colour scheme enriches the visual experience, enabling the user to quickly detect problematic situations.

The production manager can see further details and the evolution of a given KPI over time, by opening a new UI perspective, as shown in Fig. 4.11 for the Takt Time and OEE indicators of the "*Pacemaker*" station. The plotted curves are annotated with relevant information for a certain point in time, namely the current actual value and the limits on control charts at $\pm 1\sigma$, $\pm 2\sigma$ and $\pm 3\sigma$, highlighted with a visual color schema.

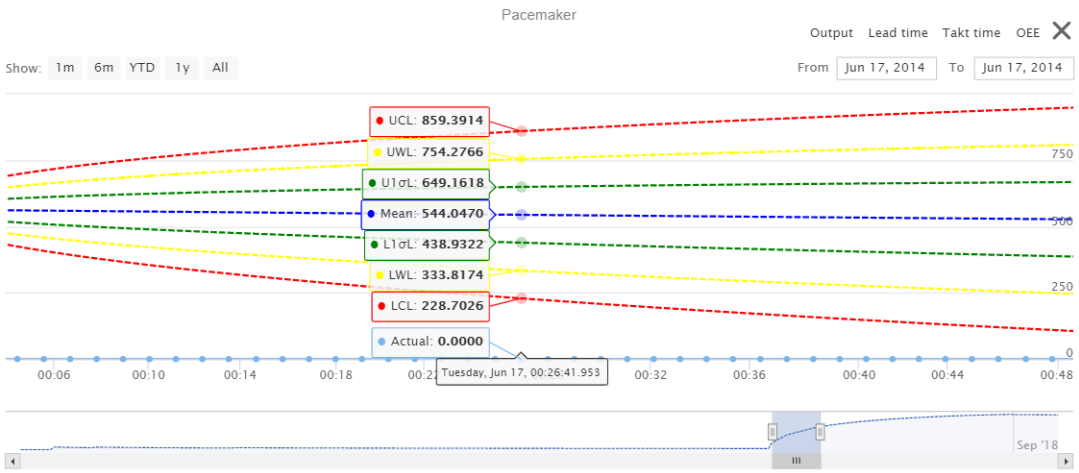

Figure 4.11: Visualization of the evolution of the KPIs along the time [111].

In this regard, the rules have been implemented in the WtR module that each agent has. The ability of the agents to collaborate allows relating various measurements and activations of the rules to extract more valuable information.

This situation permits to construct models capable of automatically identifying patterns to predict future events, operations, machine status, in other words, to detect events that constitute an opportunity to reconfigure.

Anomaly detection

In addition to the decision rules, there is a vast range of algorithms that can identify anomalies, namely anomaly detection, cluster analysis-based, and structural break. The anomaly detection and cluster analysis-based methods may be useful to discover anomalies in patterns [28], *e.g.*, the detection of the anomaly illustrated in Fig. 4.12, as "Anomaly (outlier)". The anomaly detection method can be used, *e.g.*, with:

- Dataset already labeled as "normal" and "abnormal", where a supervised algorithm determines to what class a given instances belongs;
- Unlabeled dataset, where several unsupervised algorithms identify the instances that are considerably different from the others, for example classify them as outliers.

Typically, machines produce the same type of product during a specific period but may change to produce another product family. In this situation, when a change of product type occurs in production the effects can be observed by trends in monitoring service data. For example trend deviations, which can be understood as abnormal behavior. Thus, anomaly detection algorithms are not useful here, this situation cannot be ignored since it leads to a crucial side-effect is the production changeovers.

As many solutions, just like decision rules, a simple trend analysis performed by the module still rely on a functional approach rather than powerful methods like structural break or structural changes analysis [75], which can better identify shifts and products changeover looking at data with more noise.

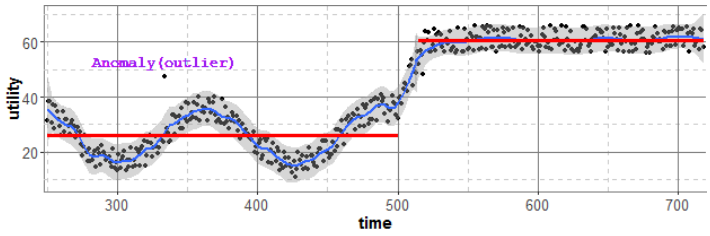

Figure 4.12: Example of trend identifying an anomaly and the reaction to a product changeover detected by the structural break analysis.

This is illustrated in Fig. 4.12, where each horizontal red line corresponds to different products). This method activates the reconfiguration trigger to proceed with the necessary adjustments for the identified upcoming batch of products. This mechanism allows us to drastically reduce or increase new production batches.

Another possibility to handle the trend triggering is concerned with the activation of the trigger for reconfiguration of services, exploring the execution of new services (*i.e.*, change the catalog of services that are offered by the agents). For this purpose, during the execution time, p, the agent is monitoring the number of requested services and the number of executed services (#Executedp) and the number of rejected services because they are not installed (#Rejectedp). In this sense, the triggering activation condition for services reconfiguration follows the equation 4.2:

$$\delta_p = f\left(\frac{\#Rejectedp}{\#Rejectedp + \#Executedp}\right) \tag{4.2}$$

where δ_p represents the rate of rejected services during the periods p. This moving average, considering the last p period allows focussing the relevance of this calculation for the last events. In general, the trigger is activated when $\delta_p > \theta$ is satisfied, where θ represents the threshold value. The determination of the θ value should be performed dynamically and at runtime and is dependent on the application scenario and system nervousness.

4.3.3 How to Reconfigure Phase

After being identified an opportunity to reconfigure, the HtR module is responsible for determining how the service reconfiguration can be implemented.

> Bear in mind that each agent of the ADvISER architecture contains a reconfiguration mechanism with a HtR module based on self-* properties. The HtR process results on a search for the "matrix of the action plan", which contains the actions to be taken, all this in an automatic, intelligent, and distributed way.
>
> The identification of who performs the actions, as well as the expected results, can be explored in two ways: selfish and competitive or in a collaborative way, as will be described later on.

Looking at the Fig. 4.13, this phase represents the second module of the service reconfiguration.

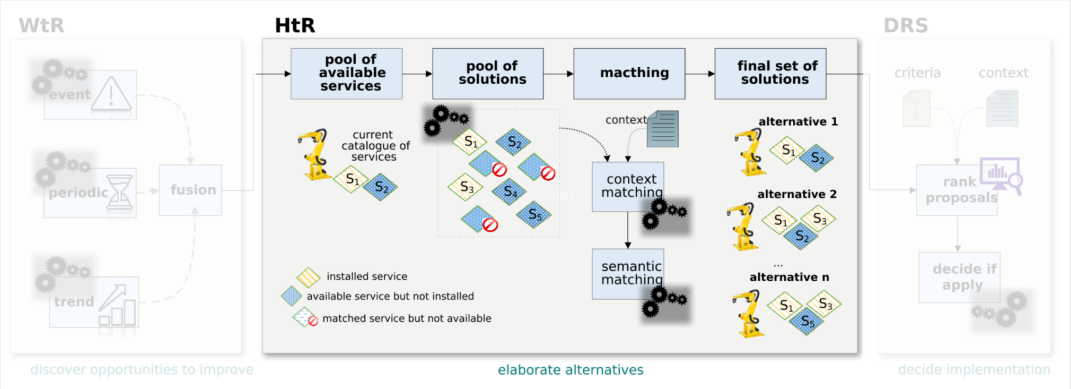

Figure 4.13: Focus on the HtR of the service reconfiguration module.

The process comprises the elaboration of a pool of possible service reconfiguration solutions, whose compatibility should be tested by using a context and semantic matching to reduce the dimension of the alternative solutions.

In general, the process of elaboration of solutions always considers the improvement of the usefulness of the resource through three classes of reconfiguration of the service (*i.e.*, to *improve the behaviour* of the service, to *change the catalog* of the service and to *change the composition* of the service). In this way, there are some change actions (*e.g.*, re-parameterization, rewiring, re-instantiation, and finally relocation) that must be considered in the implementation of any strategy (*i.e.*, reconfiguration classes).

Reconfiguration Phases

The algorithm embedded in each agent to build the space of alternative solutions for the service reconfiguration is represented bellow (Algorithm 1).

Algorithm 1 tries to find all possible combinations for the catalog of services, limited to a specific value NS. By improving each identified service presenting poor performance indexes, *i.e.*, $S_i \in S_{problem}$, through the execution of a set of actions regarding the optimization of the process encapsulated by the service, *e.g.*, calibrating tools or replacing components, or through the replacement of problematic services by others that are available but are not currently installed, *i.e.*, $S_j \in S_{available}$, which are promising to contribute for the improvement of the resource performance. Each configuration solution comprises the set of services to be provided by the resource, each one represented according to the tuple $\{s_i, a_i\}$, where s_i is the service and a_i is the action to be performed during the reconfiguration (namely nothing, improve, add or remove).

Algorithm 1 How to determine the space of configurations

Input:
Sets of services associated to an agent: $\{S_{current}, S_{available}, S_{problem}\}$
Context rules: $\{C_{rules}\}$
The maximum number of installed services in the agent: NS
Output:
Set of feasible configurations provided by the agent: $C_{feasible}$

1: **for** $S_i \in S_{problem}$ **do**
2: $\quad S_{available} \leftarrow S_{available} + S_i^{improvement}$
3: **end for**
4: $C_{candidates} \leftarrow S_{current}$
5: **for** $S_i \in S_{problem}$ **do**
6: $\quad$ **for** $S_j \in S_{available}$ **do**
7: $\quad\quad C_{candidates} \leftarrow C_{candidates} \cup replaceImprove(C_{candidates}, S_i, S_j)$
8: $\quad$ **end for**
9: $\quad$ **if** $i == 1$ **then**
10: $\quad\quad C_{candidates} - S_{current}$
11: $\quad$ **end if**
12: **end for**
13: $C_{feasible} \leftarrow applyContextMatching(C_{rules}, C_{candidates})$

4.3.3.1 Creating Reconfiguration Alternatives

The How to Reconfigure module (HtR) is responsible for elaborating the set of alternatives for the reconfiguration. Similar to the strategy of selecting agent-based CPPS to minimize the invasive CPS [44] the same strategy of having a minimal negative impact is follow in creating the reconfigurations.

Considering the self-organizing principles, as suggested by Kota et al. [91], the two significant capabilities were adopted as a reconfiguration baseline, namely:

- Management entity – that means the ability to structurally add/modify/remove entities, *i.e.*, agent and service.
- Modification of the properties – that means the ability to change the behavior of the entity, *e.g.*, to optimize a tool.

On the way to perform the reconfiguration, these two main pillars are relevant for our study, affecting the agents and the services. Table 4.2 illustrates the possible types of reconfiguration considered in this work.

Table 4.2: Possible Types of Reconfiguration.

Level	Description	Implementation Effort
Service	Change the catalog of offered services by using pluggability characteristics, *e.g.*, offering a new drilling operation	High
Agent	Change the MAS structure, *e.g.* plugging a new resource station	High
Service properties	Modification of the service properties, *e.g.*, modifying the quality measurements of a specific tool	Low

On the one hand, the agent can make behavioral changes, *e.g.*, optimize the agent's services to maintain or improve the performance at the service level. These improvements usually represent a low impact on the system if we compare them with the structural modifications. The structural changes may create a more significant impact on the system as a consequence of the introduction or removal of agents and services. The reconfiguration Decision module is responsible for deciding if the reconfiguration should be implemented and which alternative is selected, taking into consideration the control of nervousness and the learning from the past reconfiguration experiments (note that controlling the nervousness prevents the system from becoming chaotic). Being able to make better decisions in the future, each agent should observe the effects and impact of the reconfiguration through its performance improvement.

4.3.3.2 Mechanism to Create Alternative Solutions

The HtR module contains a mechanism to create alternative solutions that work based on two phases as illustrated in Fig. 4.14. The first phase is responsible for the creation of a pool of alternative service reconfiguration solutions, and the second phase is in charge of testing the compatibility of the alternative solutions by using the semantic matching.

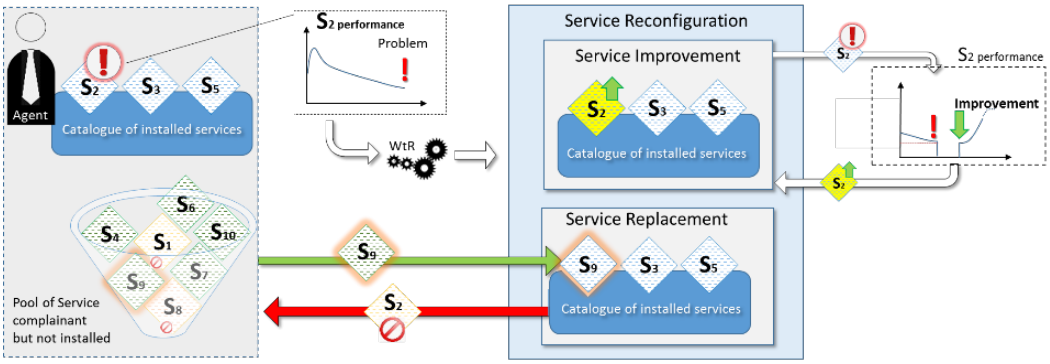

Figure 4.14: Service Reconfiguration alternatives (service replacement and service improvement).

The algorithm embedded in each agent to calculate the alternative solutions for the service reconfiguration is represented as follows:

Algorithm 2 Calculate alternative solutions

```
S1 ← select the worst service from the catalog
if ((utilization rate of Service S1 ≥ α1) OR (rate of service bids of S1 ≥ α2)) then
    Improve the Service (S1)
else
    Snew ← select the potential best Service
    if rate of service bids of Snew ≥ rate of service bids of S1 then
        Perform Service Replacement (remove S1, add Snew)
    end if
end if
```

A service with weak performance can be improved if its utilization rate or the missing bids for its usage are higher than specific threshold values ($\alpha 1$).

This can involve the execution of a set of actions regarding the optimization of the processes encapsulated by the service, *e.g.*, calibrating tools, optimizing operational parameters, or replacing components. Otherwise, the best services from the pool of available services that are not installed, are selected to create alternative possibilities to replace the one with the weak performance. Sometimes, it is useful to consider services that are not available in order to discover new opportunities for reconfiguration. The decision to explore potential solutions is calculated by the nervousness control that adjusts the threshold values (*i.e.*, $\alpha 1$, $\alpha 2$).

The learning module is capable of changing the exploration rate value, allowing to control the exploration of different solutions. This process can generate an enormous volume of service configuration alternatives, resulting in a time-consuming process. In this way, the agent can run this process in the background, mainly when the trigger for reconfiguration follows the periodic strategy.

4.3.3.3 Semantic Matching Phase

This process can generate a considerable volume of service configuration alternatives, resulting in a time-consuming task to analyze and evaluate all of them. However, some of these alternatives are not technically adequate for the current state of the system, *e.g.*, services that are not possible to be installed due to missing technical conditions. For this purpose, and aiming to reduce the space of alternative solutions, it is used a matching mechanism that analyses each solution's configuration according to the context and current situation, complemented with machine learning and semantic reasoning techniques aiming to discard non-feasible, non-reasonable and non-beneficial solutions.

Semantic reasoning tools, such as the JENA framework, permit to execute the semantical reasoning about the logical configuration and determine the feasibility of the service reconfiguration solution. In detail, as illustrated in the following rule:

$$\begin{aligned}[(?xms:\textbf{canProcess}?y) \leftarrow (?xrdf:typems:Machine),\\ (?yrdf:typems:drillService),\\ (?xms:maxDiameter?maxD),\\ (?yms:diameter?d),\\ ge(?d,?maxD)]\end{aligned}$$

This mechanism determines if the set of attributes of a proposed service matches the technical constraints presented in the resource component (in this case the diameter of the

new drill service should be less than the maximum diameter supported by the machine). Note that each service, resource, and process is semantically described in a RDF/XML format.

In the end, the outcome of this process is a set of feasible service reconfiguration solutions with the necessary correction of functional behavior but still nees to be evaluated and ranked. On flexible and reconfigurable systems, knowledge about the system is constantly changing, which increases the importance of knowledge support systems to enable decision making, especially when knowledge is distributed. The proposed architecture describes semantically the domain knowledge of the factory settings, each service, resource and process that exists in the system (*e.g.*, describing in a RDF/XML format).

The proper use of the semantic information, aside from facilitating the collaboration across multiple agents by allowing a common communication structure, it facilitates complex reasoning tasks to enable knowledge-based service composition on top of the distributed knowledge. On top of his device's information, inspired on the service description topology (*i.e.*, manufacturing-service model [70]), information in its catalogue of services (*e.g.*, gripper's characteristics), about which processes the agent can produce (*e.g.*, the resource can make process openGripper using the service gripper1) under some constraints (*e.g.*, physical limitations and QoS).

Figure 4.15 illustrates an example of an agent containing the device's information in its catalog of services (*e.g.*, gripper's characteristics). As depicted in Fig. 4.15, the agent can represent industrial robots, implementing the semantical reasoning about the logical configuration, *e.g.*, using JENA, to determine the feasibility of the service reconfiguration solution (*i.e.*, semantic matching between the resource machines and all services from the pool of services).

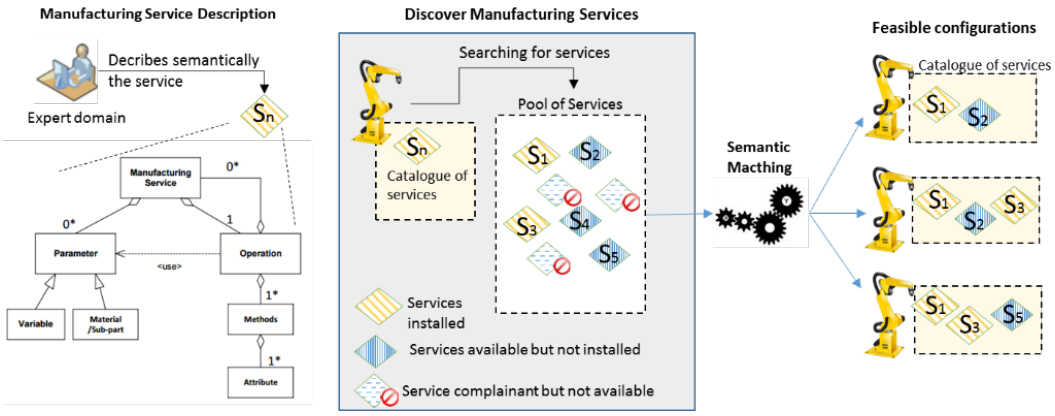

Figure 4.15: Semantic matching of service reconfiguration solutions.

The outcome of this process is a set of feasible service reconfiguration solutions.

4.4 Decision of the Service Reconfiguration Implementation

The process of selecting the optimal system configuration during a reconfiguration procedure requires an evaluation of the possible alternatives, also analyzing the viability of their implementations. This evaluation is performed in the DRS module (see Fig. 4.16).

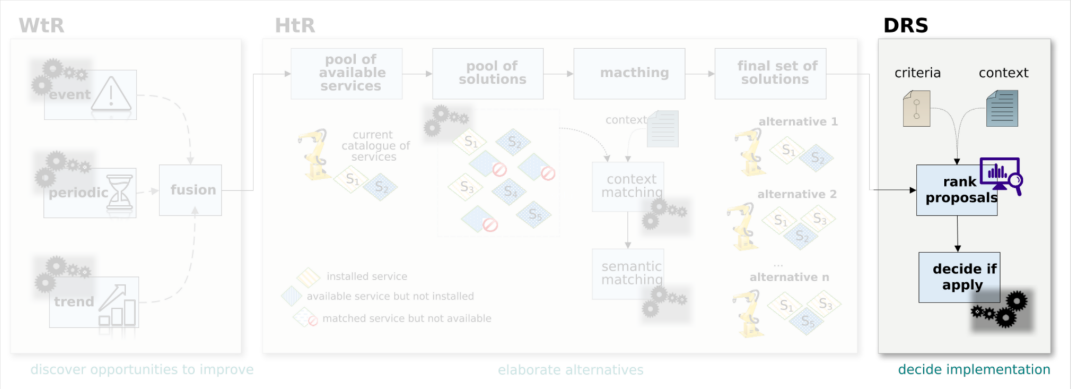

Figure 4.16: Focus on the DRS phase of the service reconfiguration module.

A goal of ADvISER is to regulate production operations either automatically or based on suggestions for reconfiguration actions for operators. In this sense, the result of this module integrates two paths.

The first part is to suggest to the operator, through a dashboard, several reconfiguration solutions in order to visualize and decide as a responsible entity of the decision whether they are valid or not. In addition, the dashboard will also work as a decision support tool to guide the operator to perform a certain reconfiguration action by using advanced Human-Machine Interface (HMI), *e.g.*, head-mounted devices. This type of tool has enormous potential in the industry as it allows training operators with little experience to produce highly customized products, performing complex assembly tasks or maintenance tasks (see more details about the term HiLCPS (Human-in-the-Loop of Cyber-Physical Systems) in order to understand more about this type of tools).

On the other hand, and from a more interesting perspective, it is considered the outcome of the DSR module in the life cycle of the agent for this, if possible, to react in an automatic mode without the intervention of the operator. Only if possible to implement the reconfiguration tasks based on self-* behaviors. This type of solution automates the process of improving operations or returning the system to normal operating conditions. The details of this module are presented in the next sections.

4.4.1 Evaluation of the Service Reconfiguration Alternatives

The agent conducts an evaluation procedure to rank the pool of possible, feasible service reconfiguration solutions provided by the HtR module according to their effectiveness.

For this purpose, a multi-criteria function is used to quantify the advantage of performing a specific reconfiguration solution, that is based on the benefits of:

- Maximizing the service composition quality, which permits to select the service with the highest quality (see Table 4.3).
- Minimizing the reconfiguration index, which will result in selecting the service configuration with best improvement values and with minimal implementation effort.

The new configuration (which should be maximized) and the cost to execute the transition into this new configuration (which should be minimized), as illustrated in the following formula:

$$score = RB/RC \tag{4.3}$$

where RB is the reconfiguration benefit and RC is the reconfiguration cost. The RB parameter is further detailed by considering the analysis of several QoS indicators, expressed in the following formula:

$$RB = \sum_{i=1}^{s} f_i(\varnothing)^{w1} \times f_i(\theta)^{w2} \times f_i(\eta)^{w3} \tag{4.4}$$

where s is the number of services considered in the configuration solution, and $f(\varnothing)$, $f(\theta)$ and $f(\eta)$ are the QoS indicators, which meaning is represented in Table 4.3. These indicators, which evolve according to the service performance and external context, provide an estimation of the benefit of the configuration solution.

Table 4.3: QoS indicators to estimate the benefit of a certain configuration solution.

Parameter	Equation	Description
Availability	$f(\varnothing) = \frac{\sum_{j=1}^{r} \lambda_j}{\sum_{j=1}^{r} \lambda_j + \sum_{j=1}^{r} \psi_j}$	Indicates the percentage of time that the service is available by considering the service uptime (λ) and the service downtime (ψ); r is the number of times that the service was executed.
Response time	$f(\theta) = \frac{\sum_{j=1}^{r} (\delta_j - \rho_j)}{r}$	Indicates the average execution time by considering the conclusion time (δ) and the requested time (ρ) for each one of the r number of times that the service was executed.
Throughput	$f(\eta) = \frac{\gamma}{t}$	Indicates the performance of the service by considering the number of times that the service was executed (γ) at a specific time (t).

On the other hand, it is also essential to calculate the cost associated with the implementation of each configuration solution, which in this work, considers the effort for the implementation of the several modifications defined in the new configuration.

For this purpose, the vector with the actual service configuration is compared with the vector with the new service configuration to determine the number of changes in the service catalog [172, 175], which should be grouped in three classes of actions: *nRS* that represents the number of service removals, *nAS* that represents the number of service additions and *nIS* that represents the number of service improvements. At the end the reconfiguration cost is calculated as follows:

$$RC = nRS \times Cost_{remove} + nAS \times Cost_{adding} + nIS \times Cost_{improve} \tag{4.5}$$

where $Cost_{remove}$, $Cost_{adding}$ and $Cost_{improve}$ represent the unitary costs of removing, adding, or improving a service.

In the end, after applying the multi-criteria function, the set of alternative service reconfiguration solutions is sorted according to the scores achieved by each configuration solution.

Reconfiguration Evaluation/Metrics of the reconfiguration phase

The process of understanding the reconfiguration effort involves knowing how to calculate the cost of reconfiguration for each potential solution, which is often ignored for the sake of simplicity. This lack of rigor in the reconfiguration process and quantification has also been identified in the literature, *i.e.*, matrix structure analysis investigates the ability to reconfigure [58]. However, the process relies on centralized decisions. This indicates a lack of research in measuring the reconfigurability effort and its impact on a decentralized way.

Mastering this challenge requires new manufacturing approaches concerning the intelligent and distributed manufacturing. Thus, the proposed model [175] inspired on [140] and [168], takes a step forward by joining the following indicators.

The reconfiguration index (RI) considers the number of reconfigurations. Each agent contains a vector with the actual configuration of the services (CC) that are being currently executed. Also, the agent contains other simulated vectors that represent the alternative configurations previously built (AR). The idea is to compare the CC vector and the simulated AR vector to understand the effort that is required [140].

$$RI = 1 - \frac{\sum_{s_i}(\text{modificationCost}(s_i))}{\#\text{ of services}} \tag{4.6}$$

$$\text{where, modificationCost}(s_i) = \begin{cases} 1, & \text{if } (cc[s_j] = AR[s_j]) \\ 0, & \text{otherwise} \end{cases}$$

According to Formula 4.6, it is created an evaluation process by the number of modifica-

tions. For example, if the alternative is equal to the current configuration, the RI is 0.

RI becomes close to 1 as many modifications exist. Besides the RI, the reconfiguration cost is another important measure, as Formula 4.6 considers the same weight for different modifications.

Aligned with the purpose of this study, other types were also investigated, the RC calculation was addressed by [226], where different service modification costs between the services are considered and calculated as follows:

$$RC = nr \times \sum_{s_i} (\text{modificationCost}(s_i) + \text{lbcost}(s_i)) \tag{4.7}$$

where nr represents the number of modifications multiplied by the unitary cost of modifying the service s_i, which includes the $modificationCost(s_i)$ of a particular service s_i and labor cost $lbcost(s_i)$.

The reconfiguration effort is evaluated with this simple metric, which considers the number of modifications required for a specific service reconfiguration. The positive impact is defined if the expected profit is higher and calculated as follows, being the Expected Benefit value calculated at the planning phase:

$$\text{ExpectedProfit} = \text{ExpectedBenefit} - RC \tag{4.8}$$

In the end, the list of solutions is ordered according to an assessment with low reconfiguration costs, high levels of benefits ensuring high quality.

4.4.2 Decide Implementation of Service Reconfiguration

After the evaluation process, the DRS module is responsible for deciding if the best service reconfiguration alternative should be implemented or not. Such a decision takes into account mainly the nervousness aspect of the entity performing the reconfiguration. In fact, such systems working in very dynamic environments, where many distributed reconfiguration procedures are performed unsynchronized and sometimes simultaneously, may lead to an unstable behaviour that limits the ability to predict, with accuracy, the future system landscape and compromises the service reconfiguration decision-making. This imposes a built-in stability mechanism on the DSR module of each resource agent to regulate the nervousness of its reconfiguration decision-making to avoid falling into a chaotic situation.

The stability mechanism imposes a time limit to perform the service reconfiguration and imposes a limit to the frequency of the service reconfiguration in the same context. The first point is related to the selection of the most valuable action, and the second one

requires an online learning algorithm to adapt the threshold parameters under a particular context, *e.g.*, defining upper and lower bounds. This allows to design a proactive system that identifies opportunities for reconfiguration but is not continually performing the service reconfiguration because it may bring adverse effects and causes a performance degradation (note that a calm system, *i.e.*, a low-frequency value, is missing opportunities to improve, but a very nervous system, *i.e.*, a high-frequency value, leads to unstable and chaotic behavior).

4.5 Extension to Collaborative Scenarios

At this stage, the described service reconfiguration approach is carried out in an autonomous and self-interested way, *i.e.*, each agent is running the service reconfiguration mechanism individually and does not share its objectives with the other agents. This approach fits well with competitive environments where the several agents act individually, competing to maximize their individual profit.

However, in collaborative environments, the lack of regulation or a normative environment using self-interested agents can lead to situations that are degrading the entire system performance, namely:

- **Deadlock situations**, where the simultaneous self-fish service reconfigurations are based on the interest of the most valuable services. This situation can lead to circumstances where no one offers the lowest profitable but necessary services.
- **Non-beneficial solutions** for the entire system, despite the benefits for the reconfiguration proponent.

In this situation, the design of a collaborative interaction protocol allows to reach a mutual agreement that benefits the collaborative system behaviour, improving the competitiveness of the system and avoiding a service reconfiguration carried out in an uncoordinated and chaotic way, and particularly preventing the existence of deadlocks [27].

4.6 Collaborative Protocols

The literature describes a variety of strategies to collaborate [199, 217]. Each strategy has a specific collaboration protocol where the goal is the same, reach a mutual agreement that benefits the collaborative system's behaviour.

Based on the available collaborative strategies a diverse set of entities might be selected. For instance, a collaboration protocol may be viewed as a set of agents and a set of services/roles which enable the agents to interact to provide a specific ability to the system.

In fact, in some of the cases a collaboration might cut across the hierarchy of the sys-

tem (*e.g.*, Product Type Agent ⇔ Resource Type Agent), in other cases the interaction is entirely horizontal (Resource Type Agent ⇔ Resource Type Agent).

For this purpose, the proposed protocol, illustrated in Fig. 4.17, considers the initiator agent as the one that wants to perform a service reconfiguration, and the participant agents as the other agents belonging to the system.

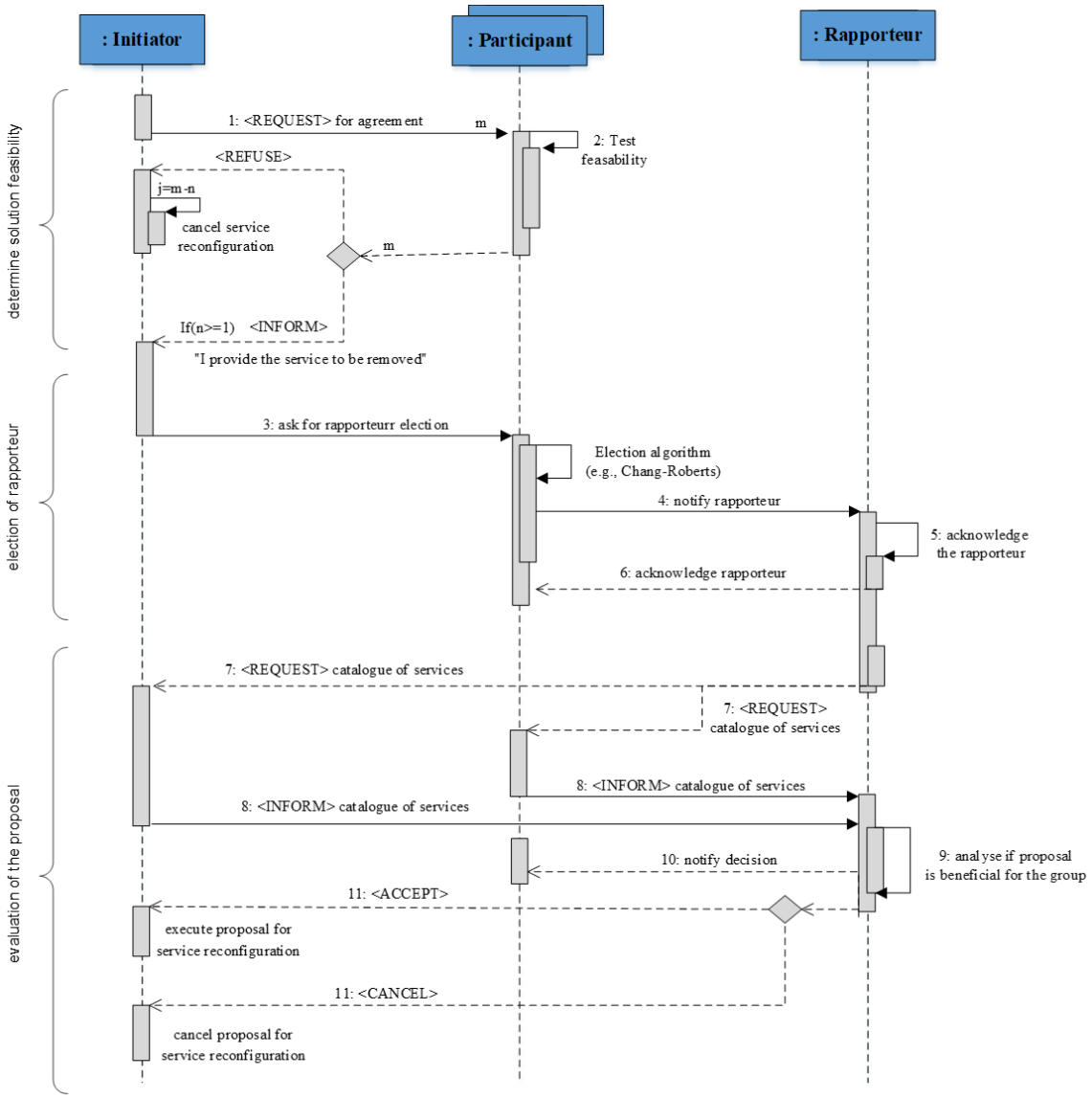

Figure 4.17: Interaction protocol to determine the reconfiguration viability in collaborative scenarios.

The initiator agent, after deciding to implement a service reconfiguration, notifies its intention to all the other participant agents by sending the "*REQUEST*" message, inquiring if someone can provide the service(s) that will be removed from the catalog of services, *i.e.*, asking the non-objection of the participants. Each participant agent will reason about its local catalog of services and will reply with "*INFORM*" " if it does not provide the

service. After compiling the replies from all participants, the initiator should cancel the opportunity to reconfigure if none "*INFORM*" message is received since despite being beneficial for the initiator, the proposed service reconfiguration solution leads to a non-feasible configuration (in the collaborative perspective). On the opposite, if at least one "*INFORM*" message is received, which means that a feasible collaborative reconfiguration solution was achieved, since at least one participant offers the service(s) that will be removed, the initiator should proceed to the second phase of this protocol that is related to the election of the rapporteur, elected from the group of participants, that will be responsible for analyzing if the service reconfiguration proposal is beneficial for the entire system (e .g., by using simulation techniques). In case that the reconfiguration proposal is found beneficial for the whole collaborative system, the initiator gets the green light to proceed with the implementation of the service reconfiguration proposal; otherwise, the initiator should cancel the intention to reconfigure. An essential assumption in this collaborative mechanism is that all agents present in the system are acting in good faith, which is expected since it is a collaborative environment.

4.7 Summary

This chapter explored the reconfiguration module to be used by each entity of the ADvISER's architecture, focusing on the service reconfiguration mechanisms, divided to work at runtime and offline. The most relevant outcomes are briefly stated below:

- Service module decisions depend on the established settings, namely when and how to apply the changes.
- All the industrial requirements and reconfiguration features discussed in the previous chapter were considered.
- A mechanism that automatically generates planning proposals was designed in a continuous manner based on what-if simulations.
- The typical autonomic model was extended to aggregate the proactive needs, enabling it to act in two operative modes:
  - online, to respond to critical events;
  - offline, to find better alternatives to non-critical events.
- Different strategies were defined to trigger the reconfiguration processes at runtime:
  - first the reactive behavior that deals with unexpected events;
  - secondly the preventive behaviour consists of an earlier detection of an event;
  - and finally the periodic behavior promotes new reconfigurations' opportunities.
- Two different approaches for the ADvISER reconfiguration were presented, namely the competitive (as default) and the collaborative.

In this chapter were described important features when engineering a flexible and reconfigurable architecture, thus approaching the thesis RQ2. However, greater focused was given to the mechanism for the implementation of the reconfiguration activation policies and the respective timings, attempting to answer the RQ1.

Chapter 5

Practical Implementation and Validation

"In theory, there is no difference between theory and practice.
But, in practice, there is."

Jan L.A. van de Snepscheut

Through this thesis it has been proven that it is possible to design an Industry 4.0 intelligent framework capable of changing the way production operations are managed today. However, we still need to confirm this thesis hypothesis: *If each computational entity of a system could regulate itself at key moments, then it would be possible to solve reconfiguration problems and promote improving opportunities in a more flexible, reliable, dynamic and intelligent*.

This chapter will focus on this hypothesis, using the framework and intelligent systems-based architecture previously described to regulate the adaptation and evolution of a system. We will start by defining a method to validate the work, in different situations, and then evaluate the obtained results. This will allow to conclude if the research questions are being met and subsequently enable the analysis of the formulated sub-hypothesis (see Section 1.3), to assess if ADvISER and its components comply with our expectations.

5.1 Case Studies

In applied disciplines, case studies are a research method essential to create new knowledge, acting as a practical proof-of-concept where the research questions [47] are addressed, and the feasibility and practicality of the elaborated approaches are tested. The applicability of the results obtained under the scope of this thesis was verified in two case studies, which had distinct Industrial applications domain levels. One case study was performed in a FMS domain, as a lab-scale demonstration, while the other was defined within the scope of one European project. By testing different Industrial applications,

we intended to demonstrate the generic features of ADvISER, making it a useful and potentially widespread tool across the intelligent manufacturing field.

Each case study grants the chance to test and enhance the ICPS feasibility, namely the flexibility required for the reconfiguration and the opportunities for improvement or change (*e.g.*, traditional production systems), in order to successfully integrate and validate ideas.

Like in any other research field, only a small percentage of the laboratory work is absorbed and adopted by Industry. However, if Academia contributed with more real-world and mature solutions the odds of having companies absorbing their research work would most certainly improve. As described in (Section 2.7), one way of measuring the research maturity is to use the TRL [54], as shown in Fig. 5.1.

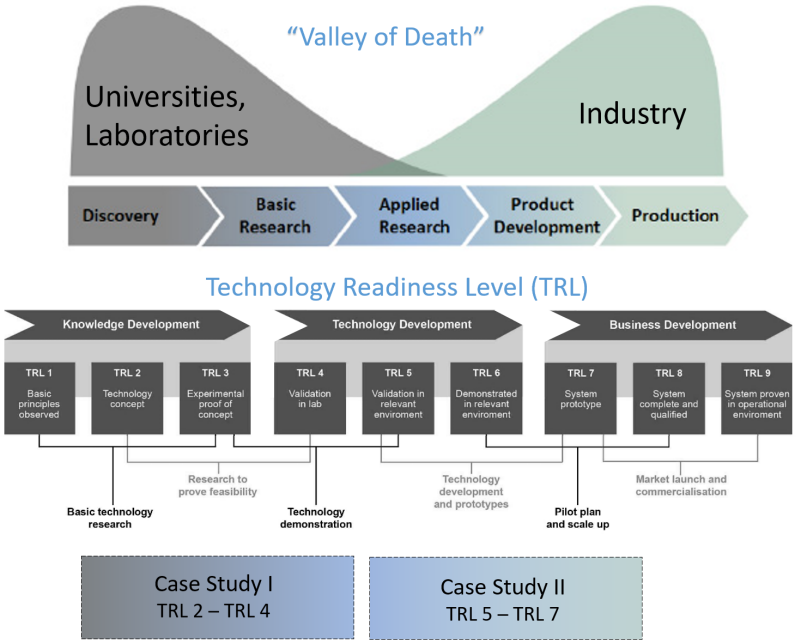

Figure 5.1: Framing the case studies at TRL and "valley of death" levels. On the top is the Representation of the "valley of death" problem in relation to the Technology Readiness Level (TRL) (Source: Own representation based on [193]).

Academic research tends to put its efforts in the TRLs 1-4 while Industry prefers higher TRLs 7-9 to work. This creates a gap between fundamental research in new technologies and their adoption by companies, which is a widely known problem called Death Valley [137]. The answer for solving this issue can be relatively simple, although not so easy to apply, since it implies increasing the maturity of the applied research, reaching midway TRLs 4-7.

The first case study aims to prove ADvISER feasibility (TRLs 2-4), testing the paradigm and technologies applied in an early lab-scale demonstration. The second case is closer to the maturity required by the industry TRLs 5-7). Therefore, it is important for this case to meet the requirements designed in ADvISER reconfiguration mechanisms and strategies.

- In the first case study an FMS scenario is explored, aiming to demonstrate the potential of the dynamic Industrial Cyber-Physical self-properties and interactions in a customized-orientated production type.
- The second case study not only allows to demystify the "software agents" topic in a real case manufacturing scenario, but also to explore Industrial Cyber-Physical Production Systems. Under these circumstances it becomes possible to, in a quality control scenario, demonstrate the importance of the reconfiguration in the regulation and improvement of the production yield based on intelligent and decentralized decisions.

In both case studies, possible implementations of Industrial Cyber-Physical Production Systems are presented and are based on the Industrial requirements. Since the goal is to validate the proposed reconfiguration mechanisms and strategies, both case studies are concentrated in the development of the ADvISER multi-agent system infrastructure and its main functionalities.

5.2 AIP-PRIMECA Case Study

The factories' complexity and the evolution they need to suffer to successfully implement new requirements involves several validation activities. These activities will allow to mature ADvISER relevance mostly by testing several strategies to enhance the adaptative and evolving cell. The cell flexible characteristics permit to implement a customized-oriented production. The FMS use case, named the AIP-PRIMECA manufacturing cell, is located at the University of Valenciennes [196]. It was essential to select an FMS since it allows to control the various processes repeated by the machines. ADvISER architecture works as a production control software, having the capacity to control the production services (adapt and evolve) to face subtle dynamic changes over time. Finally, this section entails the discussion of the several experimental results achieved by testing "job shop"-like scenarios to prove its applicability and robustness in normal and highly dynamic conditions.

5.2.1 Problem Statement

The AIP-PRIMECA FMS comprises a set of 6 workstations (WS), interconnected by a conveyor system, as illustrated in Fig. 5.2) (see [196] for more details). Each workstation offers a catalog of services, *i.e.*, a limited set of operations that can perform, being each operation offered as a service, as illustrated in Table 5.1.

The catalogs have a maximum of two services for M2 and M3, and three services for M4. As an example, the "*I_comp*" service can be executed by the workstation M4 and the "*L_comp*" service can be executed by M2.

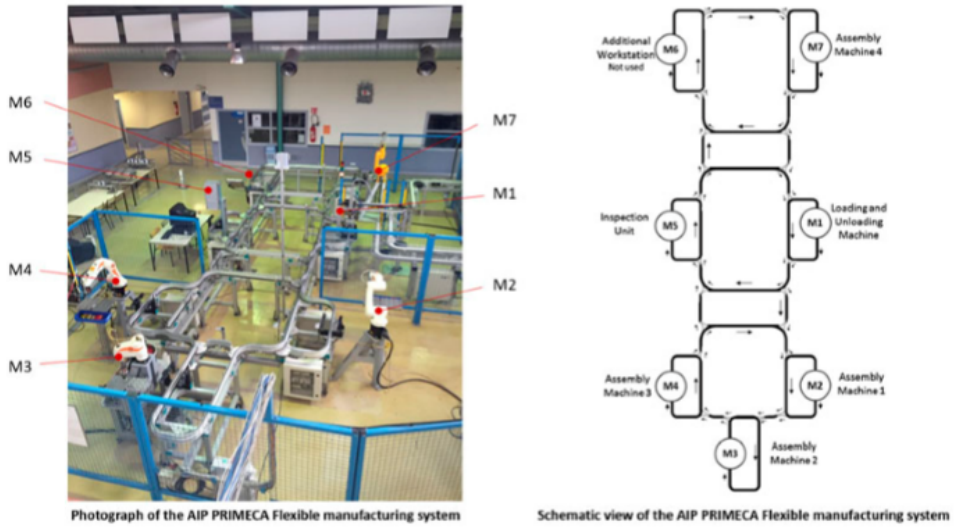

Figure 5.2: AIP-PRIMECA flexible manufacturing system (Source: based on [196]).

Table 5.1: Catalogue of services provided by each workstation (processing time in seconds).

		Workstations				
Service		**M1**	**M2**	**M3**	**M4**	**M5**
Loading	S1	I (5)				
Unloading	S2	I (5)				
Axis	S3		I (10) I (20)	I (20) NI (20)		
r_comp	S4		I (10) I (20)	I (20) NI (20)		
I_comp	S5				I (20) I (20)	
L_comp	S6		I (20)		NI (20)	
Screw_c	S7			I (20)	I (20) I (20)	
Inspection	S8					I (5)

Legend: I – offered in service catalog; NI – available but currently not offered in the service catalog.

As described on the table, some services are available but are not currently offered in the machines' catalog, *e.g.*, M4 has the "*L_comp*" service, but it is not available currently (*i.e.*, "NI (20)"). In these cases, the system can identify the benefits of offering this service and consequently can execute a service reconfiguration process to change its catalog of services and start offering this service.

As expected, this change involves some costs, namely the cost associated with the reconfiguration intervention time (usually, the reconfiguration of the catalog of services requires the stoppage of the equipment). For this purpose, the intervention time is the same for all machines, but different in terms of required intervention action:

- The intervention to improve the service performance will take 50 seconds.
- The intervention to change the catalog of services by replacing services will take 30

seconds for each service replaced.

Just as in the real-world any change requires a time to take effect, even if that time is short, so it was considered the time to complete the change, making it a case study even closer to reality. However, times in the reconfiguration process are considered for situations during execution and for situations where inactive (*i.e.*, offline mode). Of course, in offline situations, time will take a perspective of instantaneous change to the system.

The system is able to produce a set of sub-products, namely the parts in the form of the letters **A**, **B**, **E**, **I**, **L**, **P** and **T**, which combining can produce the final products **BELT** and **AIP** [196]. Figure 5.3 illustrates some examples of how these parts are assembled, and Table 5.2 describes the necessary process plans to assemble each product.

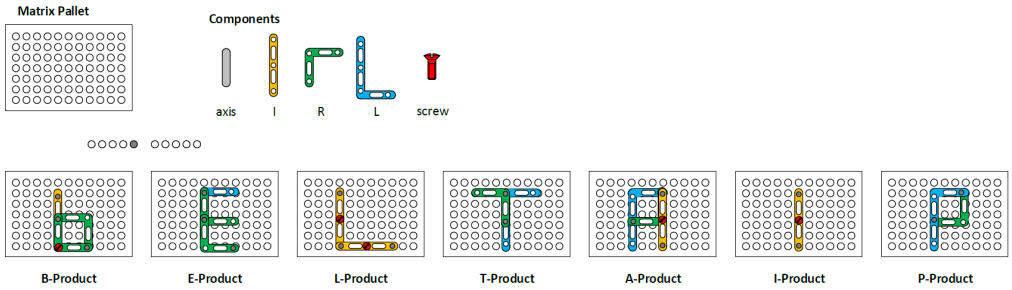

Figure 5.3: Product catalog.

The complete execution of each product requires the execution of a sequence of operations defined in its process plan, as described in Table 5.2 (for more details see [196]), which is performed by the different workstations according to their catalog of services.

Table 5.2: Products process plans (adapted from [196]).

Operation	**B**	**E**	**L**	**T**	**A**	**I**	**P**
#1	Loading	Loading	Loading	Loading	Loading	Loading	Loading
#2	Axis	Axis	Axis	Axis	Axis	Axis	Axis
#3	Axis	Axis	Axis	Axis	Axis	Axis	Axis
#4	Axis	Axis	Axis	Rcomp	Axis	Icomp	Rcomp
#5	Rcomp	Rcomp	Icomp	Lcomp	Rcomp	Screw	Lcomp
#6	Rcomp	Rcomp	Icomp	Inspection	Lcomp	Inspection	Inspection
#7	Icomp	Lcomp	Screw	Unloading	Icomp	Unloading	Unloading
#8	Screw	Inspection	Screw		Screw		
#9	Inspection	Unloading	Inspection		Inspection		
#10	Unloading		Unloading		Unloading		

The proposed approach for the service reconfiguration was tested by emulating the processes of the AIP-PRIMECA flexible manufacturing system [196].

5.2.2 Implementation of the ADvISER Multi-agent Service Reconfiguration Solution

The proposed agent-based approach for the service reconfiguration was implemented using Java Agent Development (JADE) framework [15], and following the FIPA specifications. A service layer creates an abstraction to real physical devices, where the virtual environment replicates each workstation emulating all the functionalities of the AIP-PRIMECA case study for running the experimental tests. From all the agent's types, the RAs are the only capable of performing the manufacturing functionalities without to know and control their specific implementations thanks to the capability of encapsulating operations (*e.g.*, local controller) and tools as services (*e.g.*, real devices).

The communication among the agents is performed by using FIPA-ACL compliant messages. Several agents were launched to represent the workstations, and the products requested to be manufactured in the system.

Ontology

The interaction among individual agents is crucial in MAS applications, requiring that the agents must understand each other to share knowledge, and use a proper agent communication language and proper knowledge representation. The solution is to use ontologies (as described in Section 3.6.4).

This aspect endorses as the de-facto standards for fast and, above all, easy integration of any new functionality (*i.e.*, service) in the proposed manufacturing software solution. A service layer creates an abstraction to real devices and virtual resources. In this way, the virtual environment replicates each workstation emulating all the functionalities of the AIP-PRIMECA case study for running the experimental tests.

5.2.3 Implementation of the ADvISER Virtual Execution System

The ICPS agents can aid the human along the time with better decisions in many different ways. The system is capable of performing modifications at the software level or recommending to the human workers for the best reconfigurations actions. In either case, is expected to improve the agent's expertise after each experiment and result in more accurate results. To make this possible, it implies to put the system interacting with the real-life tasks to learn faster, in other words, teach machines through trial and error in a complex (or physically dangerous) process. Practical applications using learning in the industry are increasing, but putting agents controlling and interacting with

the real production environment to learn which is the best reconfiguration actions under specific scenarios, requires to reproduce the same conditions for several experimental tests.

A useful strategy is to use virtual execution systems (VES) to simulate real system environments. This type of system lies in the ability to test different scenarios, abnormal conditions or reinforcement learning (*e.g.*, trial and error) tests in a comfortable and safe virtual world. The VES contains all the processes and resources properties to behave like a real system without the need to use the (real) physical equipment's. The implementation of this platform forced an additional effort in this work, but it was crucial to illustrate the applicability of the ICPS in Industrial contexts, enabling a prototyping and rapid test environment. However, at the same time, the VES can be considered as a cloud-based dedicated server. For this reason, the emulated models run in the server and therefore, there is no extra computation cost over in the system that is processing the ADvISER. In this way, a generic VES was designed allowing to emulate the system processes and the system behaviour (see Fig. 5.4).

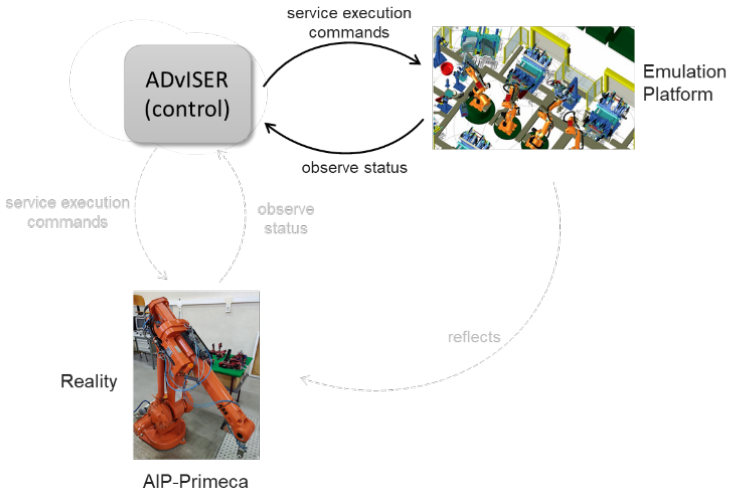

Figure 5.4: Virtual Environment System.

The ADvISER can control the virtual environment by invoking the services and observing its result, all services invoked by ADvISER for the emulation platform are indistinguishable from the VES or the cell in the real world (*i.e.*, AIP-PRIMECA).

The most important result is that this strategy allows using functions used in the case of real study, in the limit, it will be possible to be connected to machines belonging to the VES and machines of the case study, both of which are controlled by ADvISER. All the dynamics created with the VES allows the realization and validation of various scenarios:

- Disturbances (add / removal of services or agents)
- Production ramped-up
- Integration of services change-over

The tests will always be limited to the flexibility of the virtual environment. On the other

hand, the assumptions created referring to inserting new devices (*e.g.*, plug'n'produce), with this strategy is significantly more straightforward, allowing a natural way without compromise the control system offering a first attempt at a specification of a virtual system as a platform to test the regulation of the manufacturing operations most efficiently.

5.2.3.1 Implementation Prototype Development of Dashboards

One of the concerns in the roadmap for the Factories of the Future [55] consists of include the human being. In the search to include the human being, a question arises about how to draw a workplace to share the human with the different levels of automation needed to cope with the reconfiguration of processes. Therefore, considering that there are automated tasks that are too complex for the worker to manage effectively, it is practical to consider that the human can observe the production of a more abstract level and intervene if necessary.

As the volume of production data in smart manufacturers continues to grow, new methods need to consider reducing information and showing it directly to workers. Thus new ICT systems play an essential role in supporting intuitive and user-friendly user interfaces to be attractive to human workers [174].

According to the *Feature-8* from the ADvISER framework, interfaces format (*e.g.*, UI standards such as HTML 5) are necessary to enhance the visualization of complex manufacturing and production data. The dynamic dashboards were designed to explore the potential of the cross-platform technologies that deliver a seamless UI across all devices using the state-of-the-art UI libraries:

- *Bootstrap 4.0* a library that allows layout components to adapt to every single device and screen, keeping the integrity of the layout across all platforms.
- *Vue.js* a front-end framework that implements reactivity on otherwise static data that needed to refresh the page in order to fetch data.

This way data can follow to the dashboard dynamically and in instantaneous without needing to refresh the page in any device (*e.g.*, at the factory can be visualized in any typical HMI or visualized in any mobile device without changing any line of code).

One major concern from the beginning was to avoid additional computation effort in the ADvISER architecture, so as not to compromise system functioning the Industrial CPS the control should never be dependent on the graphic part. The design of this UI based on services allow to plug'n'play interfaces components making the software more flexible, easier to use and maintain.

In Fig. 5.5, is represented the designed and implemented a graphical component to reflect the logical and data layer of the agents permitting visualization of production processes,

data production, and past events during the execution of reconfiguration operations.

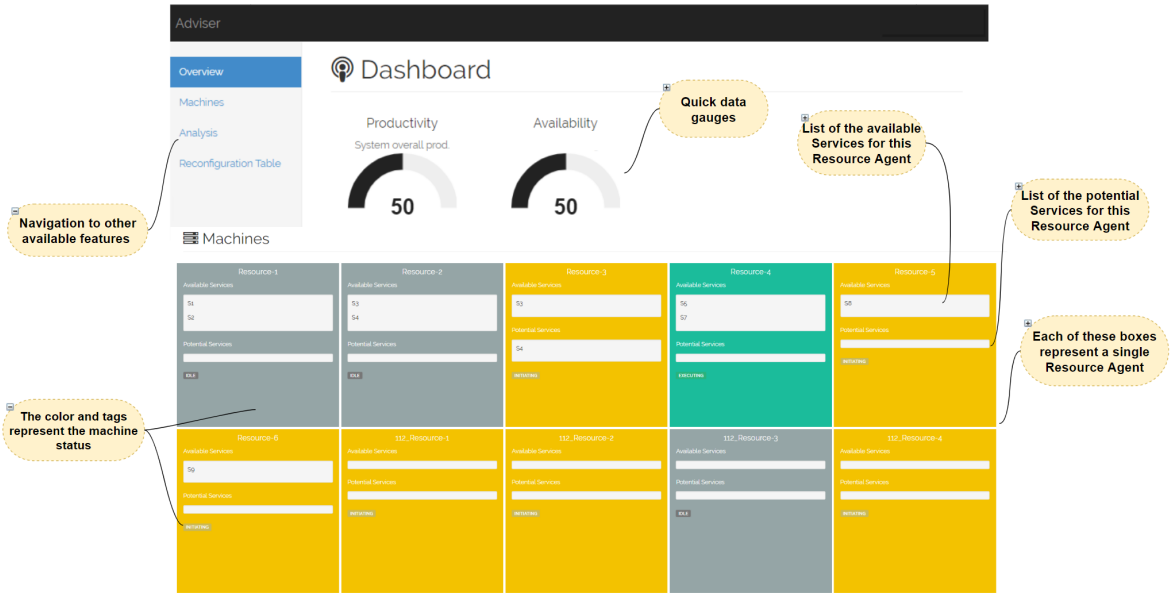

Figure 5.5: Screenshot of the ADvISER Dashboard.

The dashboard, besides, to be useful to provide a global perspective of the entire production system it allows to validate the platform in production mode, verifying its speed in the collection and analysis of the data. ADvISER dashboard permits to filter the production information almost instantaneously. For this scenario, there was effectively one objective to concentrate on all activities and in an efficient manner and see the bigger picture of the production. The global perspective contains small UI components that can be added, depending on the role of the human workers.

5.2.3.2 Dynamic Gantt

The Gantt chart illustrated in Fig. 5.6 is an excellent way for human workers to visualize the production such as service to be executed, resource status and their dependencies, as well as service and events in relation and to time creating visual scheduling.

The task assignment process is an important part of the coordination and assigning tasks in ADvISER ecosystem (*i.e.*, heterogeneous agents) is not straightforward. Since the RAs have provide different services, a protocol based on the Contract Net Protocol was used find the right agent for the right service in a distributed way. After the automated negotiation systems each agent publish its own local prespective in the dashboard. Each bar of the timeline represents a service from a specific a Product Agent allocated to a specific Resource Agent (*e.g.*, Resource agent 1-5), displaying realistic time frames particularly the start and finish date. The human worker can, in this way have a feeling of the production progress, knowing in advance whether the is necessary some intervention or whether the execution will be successful. The Gantt chart can dynamically update not just service execution status (*e.g.*, in execution, failure) or but also the time tracking.

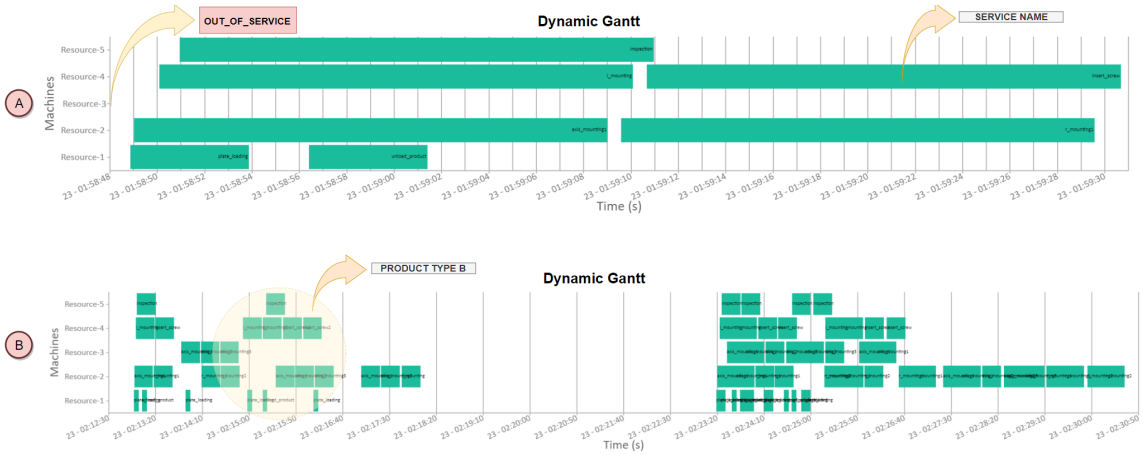

Figure 5.6: Dynamic Gantt from the ADvISER Dashboard. On the generated graphs, the selected machines are represented on the X-axis whereas the information about the service execution order is represented on the Y-axis. Note that in "graph A" the Resource-3 is Out of Service, consequently there is no services allocated to this RA. In "graph B" it can be observed an introduction of different production plans, for example, in this case the Product of type "B".

The dynamic ADvISER dashboard uses the services to retrieve the necessary data, and any slight change made during the production cell is also simultaneously reflected in the dashboard. This type of feature provides to the user a better, more accurate production organization in real-time.

5.2.4 Distributed Data Analytics

The web component is a platform-agnostic solution to illustrate several components (services). The solution, despite the performance analysis it enables functionalities that are accessible through an open API. This type of structure allows the configuration and deployment of various analytical scenarios of data on an Industrial scale.

5.2.5 Assessment of ADvISER

During the execution, the system proposes several modifications this type of work that can suffer from continuous modifications, is preferable to be assessed through emulation option, which allows:

- More flexible validation since no warm can be produced to the real system;
- Run indefinitely the ADvISER system dealing with the different scenarios:
  - Disturbances (add/removal of services or agents).
  - Production ramped-up.
  - Integration of services change-over.

Other options of assessment, like mathematical formulation or real case experiments, will not allow demonstrating so many features of ADvISER. To test and verify this model an emulation layer was developed.

Each one of the agents representing workstations has controlled by the RA, with the WtR module to identify opportunities to reconfigure, which allows considers the periodic and trend strategies.

The trend strategy uses statistical, and machine learning (ML) techniques that keep track of the performance of services and allows predicting anomalies and triggering reconfiguration opportunities. The statistical technique permits to deal with the temporal continuity of data, being used by the R "*AnomalyDetection*" library [158].

This package implements the seasonal hybrid ESD (S-H-ESD) algorithm that takes into account the trend and data seasonality, which permits to detect both global as well as local anomalies, like structural breaks, in the time series data. The ML technique used was the unsupervised K-Means Clustering algorithm to detect the outliers. In particular, after performing the cluster analysis, the instances that do not belong to any cluster are potentially indicated as anomalies, *i.e.*, those with most considerable distances to the cluster center. In both cases, the agents access to the R libraries [158] via RServe API connected by TCP/IP that acts as a back-end for services.

In the periodic strategy, the Q-learning algorithm [210] was used to dynamically adjust the sampling frequency to check the current service performance based on the positive/negative reinforcement feedbacks from previous service reconfigurations. After some interactions, the agent can select an optimal triggering frequency, in a given State. This strategy leads to a reduction of the event triggering strategy since the agent is executing the service reconfiguration more efficiently in a preventive manner.

In the same manner, agents incorporate the HtR module that allows determining the best strategy to reconfigure, namely generating potential reconfiguration solutions based on the weak reconfiguration type (*i.e.*, improving the service performance) and strong reconfiguration type (*i.e.*, changing the service catalog). JENA was used to implement the semantic verification of the matching between the pool of alternative configurations and the existing resources' context, allowing to reduce the number of these alternatives.

5.2.6 Testing Methodology

The proposed approach for service reconfiguration was validated based on the Bench4Start benchmarking [196], as the testing case to explore the reconfiguration strategies. With two distinct objectives:

- Validate the ADvISER approach, either in regular operation or in the presence of various types of disturbances, like broken services/resources.
- Evaluate the performance of the proposed strategies for the service reconfiguration to realize their benefits.

The triggering strategies described in Chapter 4 were validated by considering the testing

scenario represented in Table 5.3:

Table 5.3: Testing Scenario Configuration.

Parameters	Values
number of products	7
product Type	B - E - L - T
max number of services offered per WS	3
WS with reconfiguration capabilities	WS-4
Threshold variance (θ)	$[0.2, 0.3, 0.4, 0.5, 0.6, 1]$

The service-oriented MAS is initialized by comprising 6 agents to represent the WS disposed in the production system, and 28 agents to represent each product instance (*i.e.* 7×BELT= $7 \times 4 = 28$). After the collaborative services' allocation performed among the agents aiming to reach the optimal schedule for each production order, the system starts to run normally towards the execution of the production batch.

5.2.7 Interface with External Applications

All types of ADvISER interfaces described in Section 3.5.2.3 are considered in this case study, namely the interconnection among the agents, the service technology which can allow to reach different systems like ERP, MES, encapsulation of physical equipment such as robots and machines, complex functions, finally the graphical user interfaces and the integration with the databases.

5.2.8 Evaluate the Strategies

The idea is to let the agents search for strategic opportunities to reconfigure their services in an automatic mode during their life-cycle execution. In this work, the second hypotheses for the trend triggering were considering, being applied the equation 4.2 to generate the proper triggers too, if the $\delta_p > \theta$ is satisfied.

Figure 5.7 illustrates the results of the experiments in terms of the Cmax of each simulation considering a different threshold value.

Above each Cmax value in Fig. 5.7, is represented the number of reconfiguration performed during the experiment for that threshold value (*e.g.*, for $\theta = 0.2$, the system was reconfigured 4 times). Note that the analysis of the different reconfiguration impacts in the Cmax value should be made comparing to the Cmax value obtained for $\theta = 1$, which represents the simulation without performing any reconfiguration triggering.

The objective of these tests is to determine the θ value that minimizes the Cmax time (representing the total amount of time necessary to complete production batch) through the implementation of service reconfigurations. The analysis of Fig. 5.7 clearly shows that each tested θ value leads to different Cmax values, meaning that the selection of the

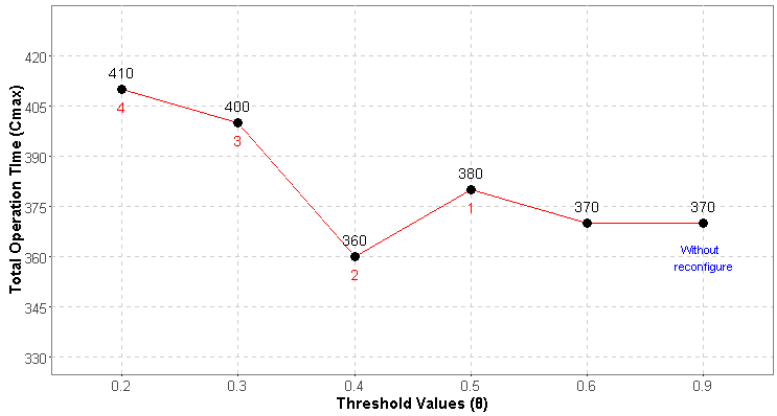

Figure 5.7: Results of applying different threshold values for the trend triggering.

θ value will have a different impact on system performance. In fact, lower values of θ, meaning that system is very nervous to constantly try to reconfigure their services, leads to the increase of the Cmax, reaching higher values than without applying any reconfiguration. This situation should be avoided in order to maintain the system stable and not in a chaotic state. On the other hand, higher values θ, meaning that the system is very nervous to try to reconfigure their services constantly, leads to an increase of the Cmax, reaching higher values than without applying any reconfiguration. This situation should be avoided in order to maintain the system stable and not in a chaotic state. On the other hand, higher values θ, meaning that the system is very calm and usually never reaches the trigger for the service reconfiguration, lead to better results than those achieved for lower θ values but slightly worse than without applying any reconfiguration.

The best threshold value of this case study is for $\theta = 0.4$, where two service reconfigurations were performed, showing clearly that the system should find an intermediate point that balances between calmness and nervousness. Moreover, it is possible to conclude that reconfiguring several times may not always be beneficial in terms of Cmax since the required time to implement the service reconfiguration will strongly affect the Cmax value. The challenge is to dynamically adjust the threshold value according to the application domain, dimension of the production batch and the impact of the reconfiguration time in the processing times.

Additionally, Fig. 5.8 illustrates the service utilization ratio for WS 4 along the time for the different threshold values.

The analysis of these results shows a considerably higher service utilization rate for $\theta = 0.4$ (which is the θ value that leads to the minimum Cmax value, as observed in Fig. 5.8 and 5.9) when compared with the other θ values. This means that the service reconfiguration considering a proper triggering mechanism represents a wiser and maximized utilization of the services.
As illustrated in Fig. 5.9, the service reconfiguration is strongly dependent on the di-

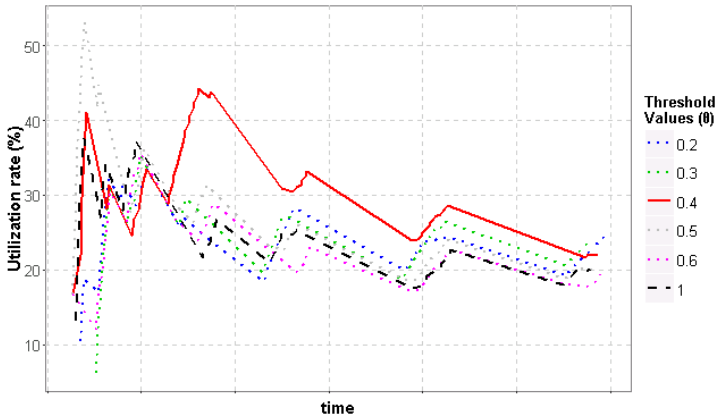

Figure 5.8: Resource utilization rate of WS 4 for the different threshold values.

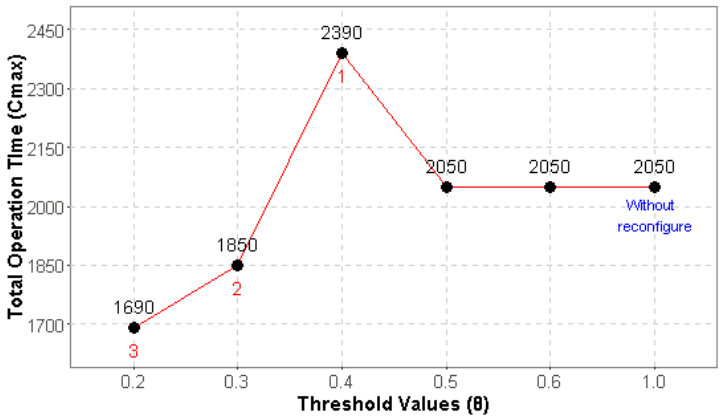

Figure 5.9: Results of applying different threshold values for the trend triggering for a larger production batch size.

mension of the production batch, having a higher impact, namely in terms of Cmax improvement, for bigger production batch sizes.

5.2.9 Critical Analysis

This work proposes an approach for the dynamic, efficient, and on-the-fly reconfiguration of services. The proposed approach actively explores and promotes different triggering strategies, embedded in smart agents, that does not only identify opportunities to change, but also to assist engineers and managers in exploring and deciding about different alternative configuration scenarios, to cope with potential disturbances or production changeover challenges or recommend service modifications.

Three different triggering strategies were implemented on the WtR, namely event, periodic and trend, and posteriorly tested using an FMS case study. The experimental results showed that the proper use of the dynamic service reconfiguration mechanism is strongly dependent on the threshold value, which should be dynamic, and on-the-fly adjusted by using learning mechanisms.

5.2.9.1 Collaboration

The experimental tests conducted in the present study compared and evaluate different batch orders of the product BELT, to test the feasibility of the service reconfiguration mechanism. In case the agents decide to perform a service reconfiguration, the physical resource is stopped during the execution of the maintenance intervention to improve the service performance (which takes 50 seconds) or to replace the provided service (which takes 30 seconds for each replacement). Aiming the simulation of realistic scenarios, it was considered that workstation M2 have a probability of failure of 4% for all services in their catalogs, staying out-of-service during 300 seconds for the execution of recovery actions. After the execution of an intervention to recover or improve the service performance, its probability of failure is reduced by 1,5%.

Next, it can be observed different batch orders, in particular, 10, 20 and 30 BELT products were simulated under three different scenarios, as described below.

5.2.9.2 The Service Reconfiguration Mechanism is Disabled

Figure 5.10 illustrates the experimental results for the differences described scenarios, considering the makespan (*i.e.*, the necessary amount of time to execute the entire batch of products) that illustrates the production efficiency, and the workload distribution (*i.e.*, the average of the standard deviation of the service utilization for each machine) that illustrates how well balanced is the production (values close to zero represent a well-balanced production).

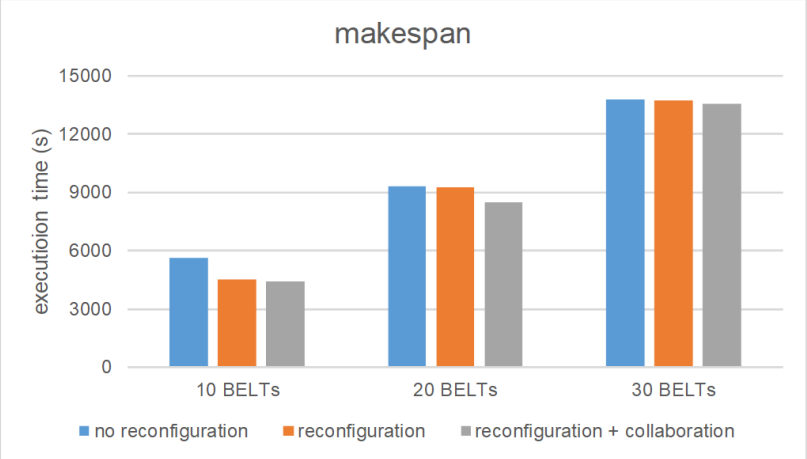

Figure 5.10: Experimental results for the lead time, considering different batches and types of service reconfiguration.

Initially, the system was running with the service reconfiguration and collaborative mechanisms disabled.

5.2.9.3 The Service Reconfiguration Mechanism is Enabled but Without Collaborative Capabilities

The second scenario considers that each resource agent is running in a selfish mode, which means that agents will execute their service reconfiguration individually and without any collaborative procedure. The achieved results show the benefits of using reconfiguration, with the best improvement achieved for the 10 BELT batch scenario (improvement of 19%), which means that for larger batches the occurrence of disturbances is more diluted and the reconfiguration has a shorter percentual impact. A more balanced workload among the machines is also noticed when applying the service reconfiguration and the collaborative service reconfiguration.

5.2.9.4 The Service Reconfiguration Mechanism is Enabled With Collaborative Capabilities

A third scenario considers that the collaborative mechanism is enabled, running over the distributed self-fished service reconfiguration process. In this case (see Fig. 5.10), the achieved results show an increase of the system performance (illustrated by the reduction of the makespan in 21,2%), with the reduction for larger batches following the same pattern as identified for the previous scenario. In the same manner, a reduction of the workload distribution is also noticed, which means a better balancing of resource utilization (see Fig. 5.11).

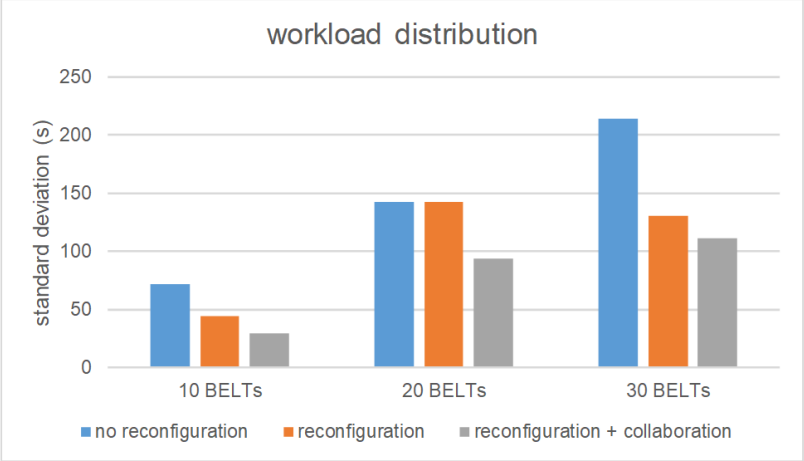

Figure 5.11: Experimental results for workload distribution, considering different batches and types of service reconfiguration.

This scenario shows the benefits of applying the collaborative interaction protocol to balance the service reconfiguration performed individually by the distributed agents. This set of experiments allowed to verify the benefits of using service reconfiguration, illustrated by the increase in the system production efficiency, and particularly the advantage of combining the collaborative interaction protocol to better balance the service reconfiguration performed individually by the distributed agents.

These results also show that the proposed service reconfiguration approach is dependent on the dimension of the batch size, the frequency of service failures and also from the recovery time, which requires the consideration of learning mechanisms to decide if the application of the service reconfiguration is beneficial or could have a counter-productive effect.

5.2.9.5 TRL Analysis

Usually, a novel idea should be object of further experimentation and testing before it can be prototyped. This case study provided a lab-scale demonstration of ADvISER and allowed to largely explore its features. So, generically, it can be positioned at a readiness level between 2 to 4. Nevertheless, the design of ADvISER was supported by several paradigms and technologies that are well-known (*e.g.*, SOA and MAS) in modern manufacturing systems. But, even if several ADvISER components are available at the enterprise level, showing a readiness level of 7 or punctually 8, their adoption is still at the early development phases. Notwithstanding, the adoption of technologies already used and adopted by the Industry is a good strategy to increase the overall TRL.

5.3 Whirlpool Use Case

The opportunity to instantiate an Industrial Cyber-Physical Production System into a real Industrial production line arises in the European Project GRACE (inte**G**ration of p**R**ocess and qu**A**lity **C**ontrol using multi-ag**E**nt technology). The project is aligned with the idea to develop a modular, intelligent and distributed manufacturing control system, using the MAS paradigm and integrating production and quality control processes.

5.3.1 Problem Statement

The problem addressed in this case study considers a factory plant owned by Whirlpool and located in Naples, Italy, which produces laundry washing machines. The production layout is composed of several workstations disposed in sequential order, as showned in Fig. 5.12.

The pallets circulate in the production line through a conveyor system, following a process plan that must be fulfilled to produce the desired washing machine. The process plan contains the information regarding the materials' variables (*e.g.*, type of the rear tub, bearing, or drum) and the operations' parameters (*e.g.*, the thickness of the welding process or torque of the bearing insertion process) according to the type of washing machine to be produced. Each workstation disposed along the production line executes a single operation in the product. The operation can be of the following type:

- processing (*e.g.*, bearing insertion, tub welding, or pulley screwing);
- inspection (*e.g.*, gap control, assembly visual check, or functional tests);
- manual operation (*e.g.*, electronics and cable assembly).

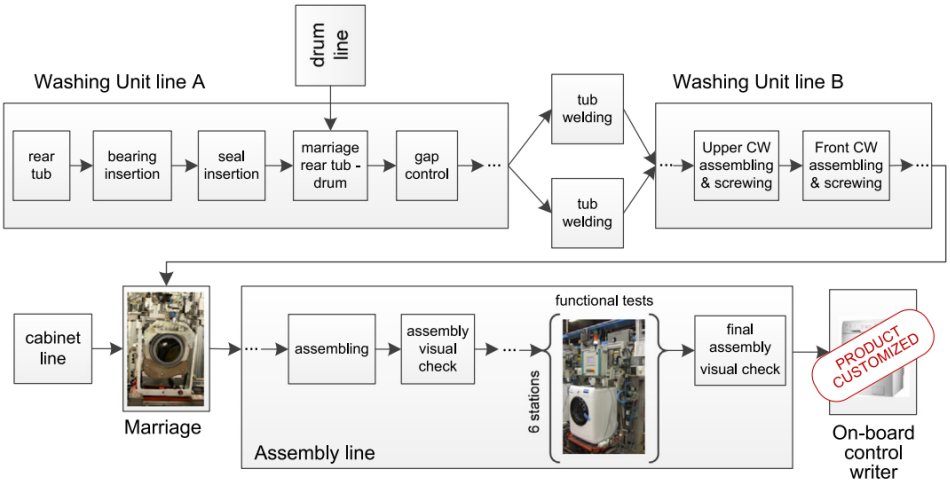

Figure 5.12: Layout of the use case production line [112].

The production execution is performed sequentially and rigid manner without the dynamic adaptation to unplanned changes, which usually occurs in such Industrial environments. Unexpected condition changes may occur in the process execution (*e.g.*, slight deviation of the quality of the tub welding operation) or in materials' characteristics (*e.g.*, deviation of the drum's speed from the expected 3000 rpm to the installed 2950 rpm). Another example is the functional test area that comprises six stations to execute a set of functional tests to all product instances produced in the production line (see Fig. 5.12). Currently, this operation lasts 6 min and comprises a fixed plan, even if some tests may be redundant according to the results already gathered from the previous inspection tests.

This work aims to integrate the process and quality control, establishing feedback control loops to support the on-the-fly adaptation of process and product parameters, to improve the product quality and production efficiency. Examples of such procedures are the following:

- Adaptation of the operations' parameters;
- Earlier identification of the quality problems;
- Customization of inspection tests, and the customization of the final product.

In order to implement a MAS in a manufacturing plant, it is necessary to associate software agents to specific stations acting on the line and performing process operations. This process implies that hardware is associated with each agent, and the device procedures can be accessed through an service interfaces, which technically represents an Industrial Cypher Physical Production System.

5.3.2 Agent-based Architecture Overview

The designed agent-based system, depicted in Fig. 5.13, comprises autonomous and cooperative agents representing the manufacturing components disposed along a production line producing washing machines [165].

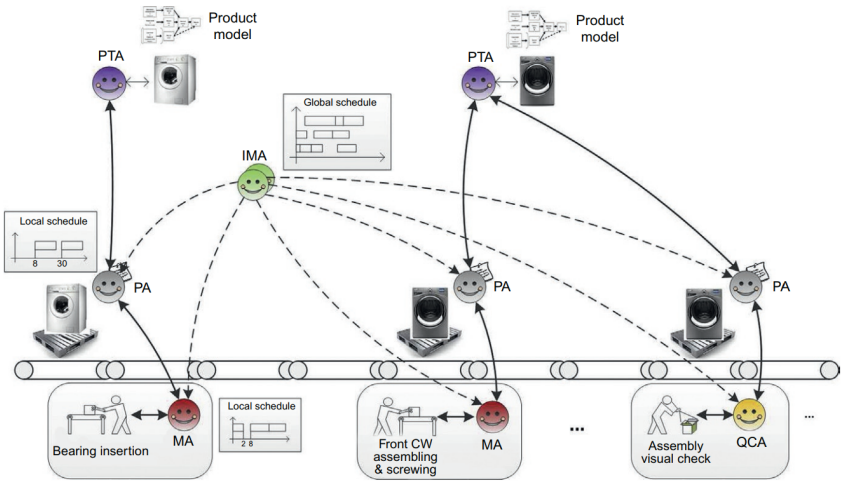

Figure 5.13: Multi-agent system architecture for the production line [165].

In such distributed environment, several types of agents were instantiated (described in Section 3.6), according to the process to control and to their specialization (see [107], for more details):

- Product Type Agents (PTA) representing the catalogue of products that can be produced by the production line. They possess the knowledge required to produce the products;
- Product Agents (PA) handling the production of product instances along the production line (e.g., washing machines and drums). They possess a process plan to produce the product and interact with the agents responsible for the process and quality control;
- Resource Agents (RA) representing the physical resources of the production line. The RA comprises several specializations according to the particularities of the resource, namely Machine Agents (MA) and Quality Control Agents (QCA);
- Independent Meta Agents (IMA) implementing global supervisory control and optimized planning (this agent is optional, i.e., the system can continue working without them, however losing some optimization).

Briefly, the PTA receive orders from the ERP/MES system and launch PA to execute the production requests, exchanging product and process planning information. The PA interact with the RA and QCA during the execution of the process plan, re-routing the pallets according to the current production line conditions and querying about the progress of the plan execution. The PA, RA and QCA interact with the IMA to provide feedback information about the production execution and to receive optimized guidelines to improve their execution. The presence of IMA is optional and aims to provide global decision-making strategies based on data analysis methods that exploit information collected from individual agents. The RA interact between themselves during the physical synchronisation of production activities, such as for the transportation of the pallets between workstations.

These type of agents follow the principles in Section 3.6.2 [112], and are validated using a Petri nets methodology to specify the system behaviour, which take advantage of the well-founded mathematical theory that allows to design the control system behaviour, but also validate the system specifications, supporting the design-implementation path (available on [108]).

Interaction patterns are required to model the cooperation among agents and to coordinate their actions to produce a product, enhancing the integration and adaptation of the production and quality control processes. As an example, Fig. 5.14 illustrates the interaction protocol: initially, the RA detects that a pallet has arrived at its machine station by reading the moby RFID identifier associated to the pallet, and notifies the PA associated to the pallet. The PA determines the program and the parameters to be passed to the RA for the operation execution, based on the process plan and the information about the execution of previous operations.

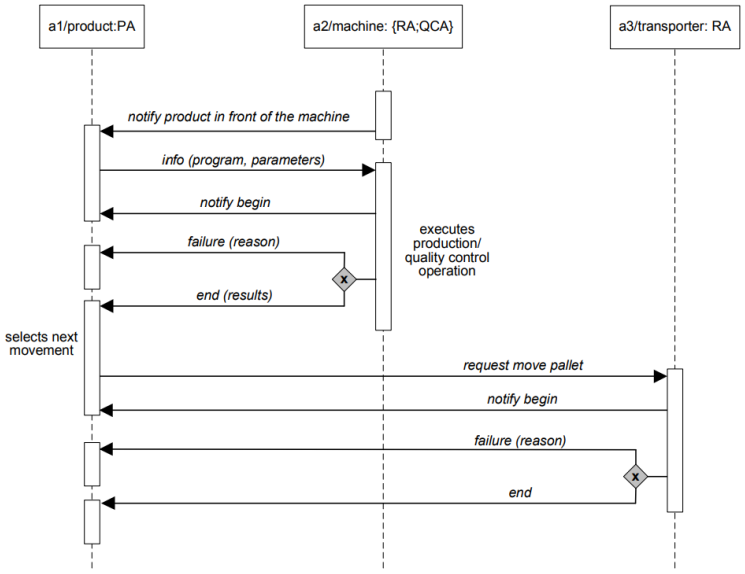

Figure 5.14: Interaction Diagram for Operation Execution Process.

After finishing the execution of the operation, the RA notifies the PA about the results of the operation execution: end if it is concluded with success or failure if any problem had occurred during the execution. At this moment, the PA determines the next operation and requests a movement of the pallet to the next station to a transport resource agent, which will deliver the pallet in the target station. The design of other interaction patterns requires more complex behaviours, and knowledge. The interaction among individual agents is similar to the previous case study (as described in Section 3.6.4). The communication language and proper knowledge representation are based on the development of an ontology for a multi-agent system controlling a production line were proposed in [115]. The ontology is generic, but if necessary, it is easily extended to support new concepts.

5.3.3 Dynamic Service Reconfiguration and Integrating Process and Quality Control

The distributed agents can integrate Production and Quality Control and implementing procedures to adapt the production process based on the dynamic service reconfiguration principles. For this purpose, self-* (-adaptation and -optimization) will consider the information collected locally by the agents responsible for the production and quality control. An example of this type of mechanisms, namely the interaction pattern to adjust the processing parameters of the machines and quality control stations based on the information provided by the quality control tests. Figure 5.15 illustrates the collaboration among the agents and the adaption mechanisms aiming to increase the overall quality and performance of the productive process [194].

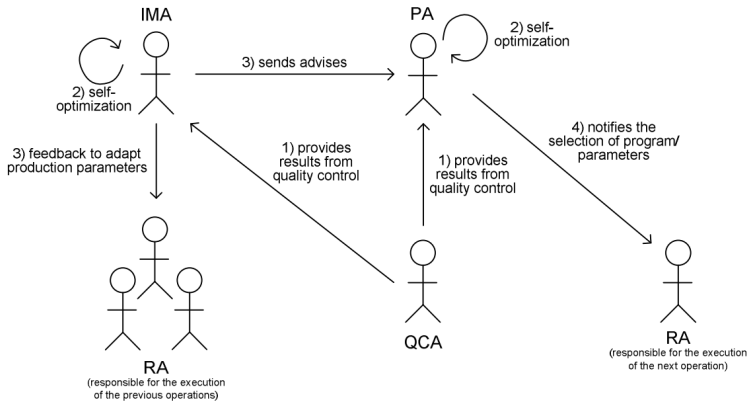

Figure 5.15: Adjusting the Processing Program/Parameters of the Manufacturing Machines [107].

- The QCA after performing the inspection tests over a product instance provides the results to the PA and IMA while running locally the online reconfiguration mechanism;
- The IMA runs an offline reconfiguration mechanism for analysing possible deviations and proposed service reconfiguration alternatives to RA;
- IMA can suggest reconfiguration opportunities in runtime to the PA in a continuous way, (e.g., new strategies and policies for the production execution). These agents may use this information to reconfigure the services to evolve towards the execution of future operations, or also to adapt the current services (i.e., correcting the detected problems/deviations);
- The PA uses the feedback information from the quality control operations to select the machining program, to adjust the parameters of the program or even to select the components to be used during the execution of the next operation. Note that the RA associated with the machine responsible for performing the next operation about the adequate reconfiguration target (i.e., program/parameters/components) to be used during the execution of the operation;
- In an online reconfiguration perspective, the RA can self-adapt the configuration of the quality control tests to be executed by the quality control stations, by selecting the proper algorithms and/or testing parameters. In this case, the PA uses the feedback

from the previous quality control tests and the advises provided by the IMA to decide about the best algorithms and/or parameters to be used in the next quality control operation by the RA.

The achievement of adaptation mechanisms encompasses a strong effort in designing proper reconfigurations mechanisms (i.e., optimization/adaptation) for each agent and in designing cooperation mechanisms to support the combination of the local and global knowledge/adaptation behaviours. It should be noted that traditional solutions based on centralized and rigid structures exhibit a weak adaptation to condition changes, in [165] it is explored the distinct levels of adaptation performed by each agent at local level, and the adaptation driven by the presence of agents with a wide and global perspective, as illustrated in Fig. 5.16.

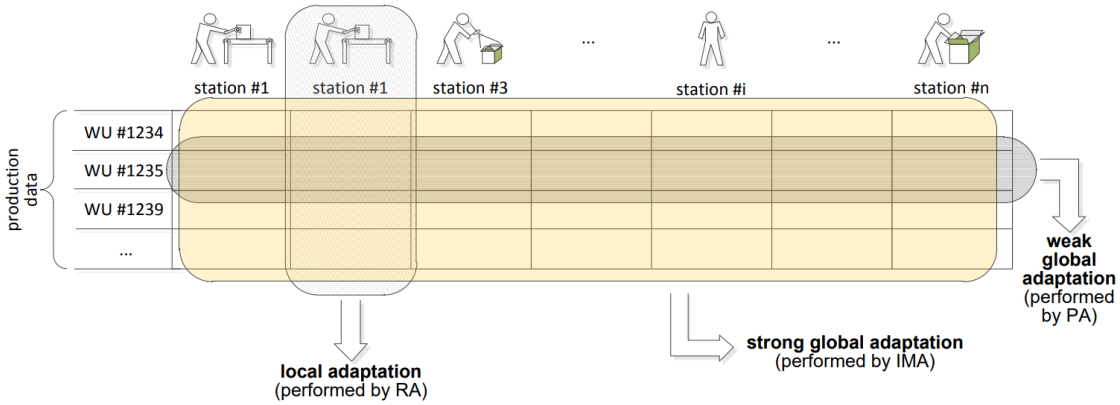

Figure 5.16: Different types of self-adaptation [165].

Namely, three different adaptation levels can be identified[165]:

- **Local self-*:** self-adaptation and self-optimization performed by one agent and involving only its own data, *e.g.*, the RA that can improve its behaviour taking into consideration the historical data about the execution of previous services.
- **Weak global self-adaptation:** performed by one agent collecting data related to a product instance or process from the other individual agents.
- **Strong global self-adaptation:** performed by one agent at *offline* mode aggregating the entire production data to detect correlations on the data.

These adaptation procedures will trigger only logical reconfiguration actions or notifications. Examples are the dynamic adaptation of the functional tests plan, the customization of the on-board electronic controller, and the generation of warnings at any point of the production line in case the desired quality is not possible to be achieved anymore.

Industrial Cooperation Patterns

The production line control system emerges from the cooperation among the agents, coordinating their actions according to the agents' goals. In Fig. 5.17 gives a real-world

perception of a CPS component working in smart production executing online and *offline* reconfigurations. The ***offline* reconfiguration** (*i.e.*, named as global self-adaptation), will be illustrated through the description of an adaptation mechanism for the customization of the functional test plans performed by the inspection station placed near to the end of the production line [165].

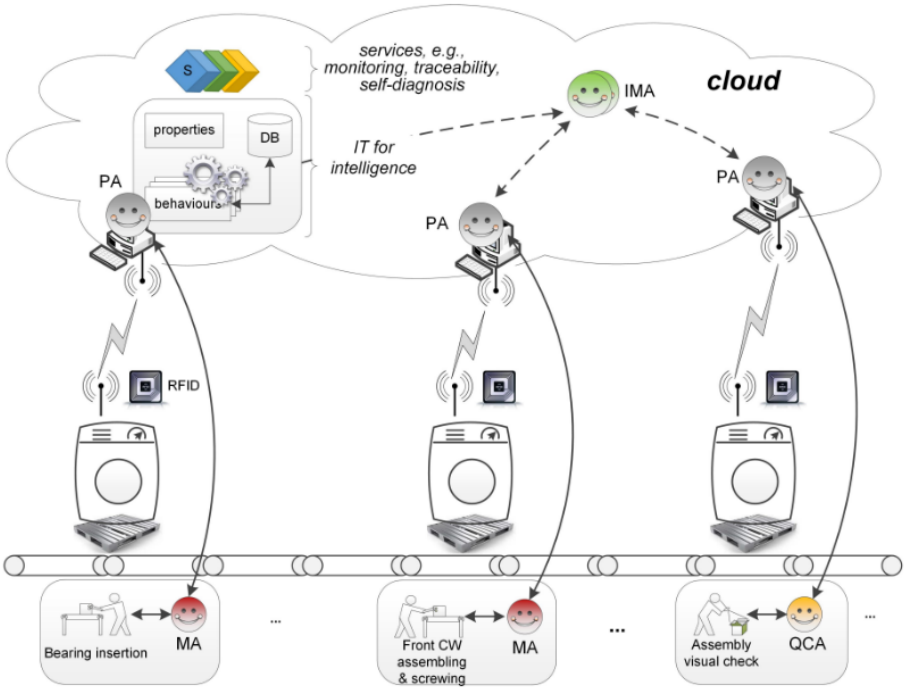

Figure 5.17: Intelligent product vision applied in the washing machines production line (production phase) [110].

Figure 5.17 illustrates where both the **weak and strong global adaptation** can be performed by the PA and IMA, whith IMA running in *offline* mode.

Along the production line, the PA is receiving the service feedback regarding the results of the execution of the processing and inspection services, respectively from MAs and QCAs. When the PA arrives at the functional tests station (see Fig. 5.18), it executes a self-adaptation service to customize the plan of tests for this particular washing machine, correlating the historical production data of the washing machine. In parallel, the PA can request support from IMAs, which, based on their wider perspective, can provide advice on the execution of the self-adaptation procedure.

Figure 5.18 illustrates one example of these interaction diagrams, showing the interaction among the agents during the process of adaptation of the functional tests plan.

5.3.4 Scenario 1 - Customization of the Functional Test

The Functional Test (FT) station comprises 6 boxes to perform a set of FT to all product instances produced in the production line. The execution of the set of tests lasts 6 minutes

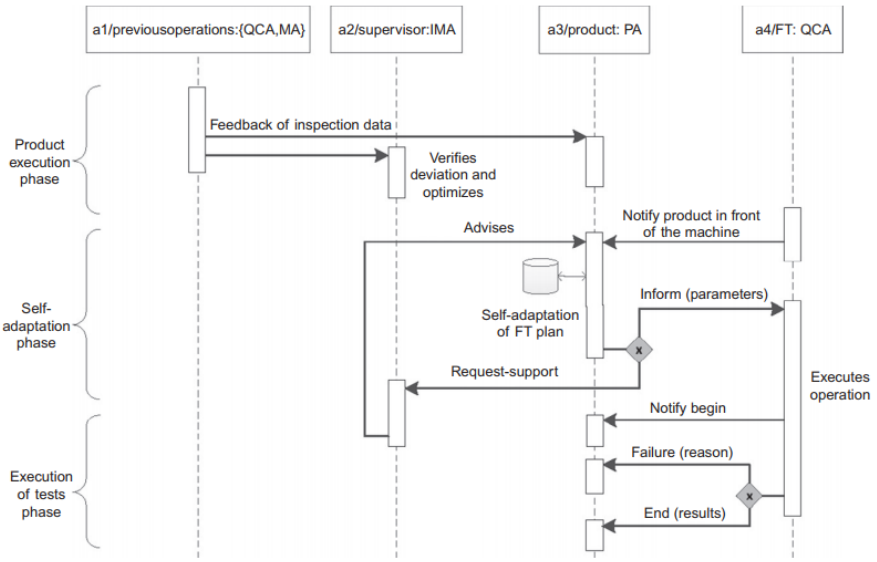

Figure 5.18: Cooperation pattern for the adaptation of the functional tests plan (from [112]).

and comprises a fixed plan of tests. The possibility to customize the plan of tests by adapting to the necessary tests, adjusting others or customizing the messages to the operators may represent a significant improvement of:

- *Product quality*, since most effective quality control procedures are performed;
- *Productivity*, since the inspection time is reduced.

The architecture of each inspection station performing the functional tests is represented in Fig. 5.19.

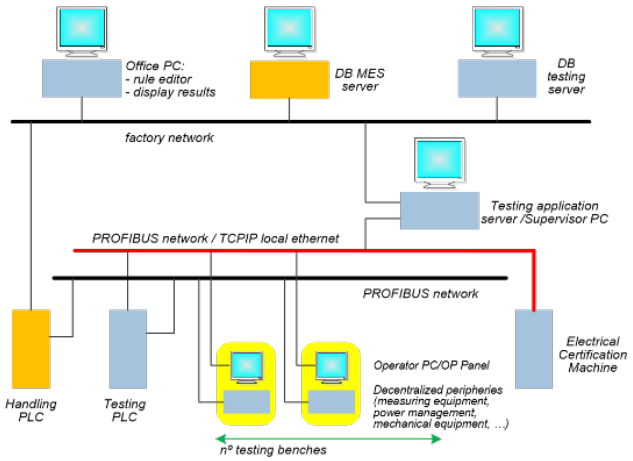

Figure 5.19: Architecture of a testing station [165].

A PLC is responsible for executing the testing rule (one specific rule for each product model). The rules and the parameters associated with these rules can be edited by using the "Testing Rule Program Editor" and the "Table of Appliance Parameters Editor" that runs on an office PC. The testing rules are stored in the "Server DB", which also stores the data results from FT.

The adaptation of the FT plan represents a situation of global self-adaptation with two perspectives: one performed by the PA related to customizing the FT plan and other embedded in IMA to tune the correlation parameters.

The idea is to use the data collected from the individual RA, related to the operations executed along the production line for a specific product appliance, to customize the FT plan accordingly.

This knowledge is embedded in the PA that will influence the FT station by customizing the default test plan, namely:

- Changing the sequence of the tests for a specific washing machine;
- Customizing the messages provided to the operator, *e.g.*, highlighting particular points for a more detailed and effective test.

The pertinent issue is the establishment of a proper adaptation function that correlates data to customize the testing rule and parameters for each appliance (in the rule editor). The next sections detail the several steps comprising the adaptation mechanism for the customization of the FT plan.

5.3.4.1 Calculation of Performance Indicators

Along the production line, the PA receives from each RA (*i.e.*, MA or QCA representing respectively processing or quality control stations) a service inspection result that consists of [194]:

- An overall result (*i.e.*, OK/KO);
- A detailed result (*e.g.*, a graphical evaluation of a parameter over another parameter);
- A performance index, *i.e.*, P_j.

The performance indicator of the process quality (P_j) calculated by the RA (see Fig. 5.20). The P_j value can be in the interval [0, 100], being 0 the representation of a bad process and 100 a good process.

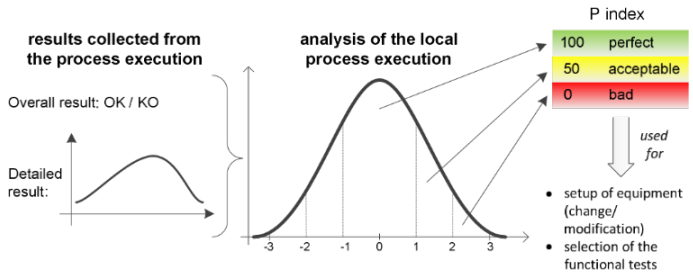

Figure 5.20: Classification of the inspection results at local level [165].

A simple example is the determination of the matching degree between the ideal and real curves (that represents the correlation between the depth and torque parameters over

the time) in the bearing insertion process, and corresponds to the likelihood of a decision (e.g., distance from the borderline in the classification space). This information can use this information to execute the adaptation procedures at any time of the production of the appliance.

5.3.4.2 Generic Testing Rule

The PA applies a function that uses the information collected along the production line, and particularly the P_j indexes, to customize the sequence of the tests to be performed by the functional test station.

A parameterized testing rule was defined to execute the FTs decided by the agents. Since the testing rule is made of a long list of "*elementary action blocks*" and the sequence follows Boolean rules conditioned by events like the success or the failure of one test/action contained in a block, it was defined conditions (using variables) to jump unnecessary tests according to the agents' decisions. In this way, if a value coming from the agents assigned to a variable is inside or outside a boundary, the test can jump on a sequence of blocks instead of others. Fig. 5.21 illustrates a fragment of the structure of the generic customized testing rule used in the agent-based system.

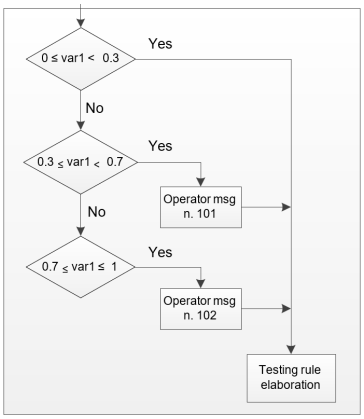

Figure 5.21: Structure of the generic customized testing rule [165].

The testing rule can be generalized to be used for all appliances, being the information about specific measures limits or dedicated variables read from the "*Table of Parameters*". In this way, the rule (executed by the testing PLC) performs measures and sequences adapted for the specific appliance under test.

5.3.4.3 MPFQ Correlation Table

The Material/Process/Function/Quality (MPFQ) model identifies the correlation among four elements [66]:

- Material: all that is required to produce a particular product or product component;
- Process: operations aiming to process and transform materials into the final goods by using machines, tools and human labour;

- Function: characteristics of a product item;
- Quality: the degree of conformance of final product functions and features to customer requirements.

MPFQ model allows correlating the material, processes and functions to the quality parameters of the product being produced (*i.e.*, the contribution of each material, process and function for the product quality indexes). Figure 5.22 illustrates the structure of the MPFQ model, where the table parameters, SQ_{jk}, represent the quality correlation index (belonging to the interval [0, 1]).

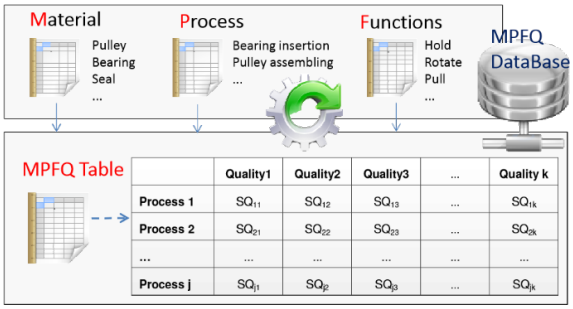

Figure 5.22: Overview of the MPFQ structure[165].

Based on the project needs, eight quality parameters are considered, (*i.e.*, Noise (Q1), Energy Saving (Q2), Component Conformity (Q3), Assembly Conformity (Q4), Off-Leakage (Q5), Washing Performance (Q6), Safety (Q7), and Green Footprint (Q8)). This metrcis will allow highlighting the contribution of any individual assembly processes and quality for the overall appliance quality produced.

5.3.4.4 Adaptation of the Testing Plan Variables

When the appliance arrives at the FT station, the PA computes the adaptation function for customizing the testing plan based on the set of variables that define the sequence of tests in the generic testing rule that is used by the PLC to control the FT station. In this case, the adaptation function uses the collected M_j, P_j, F_j indices for each process and correlates that data with the MPFQ correlation table taking into consideration the model of the appliance, as illustrated in Fig. 5.23.

The set of variables var_k, that customizes the sequence of the test plan by redirecting the flow of tests in the generic testing rule (see Fig. 5.23), is given by:

$$var_k = \frac{\sum_{j=1}^{n} SQ_{jk}.M_j}{n} + \frac{\sum_{j=1}^{m} SQ_{jk}.P_j}{m} + \frac{\sum_{j=1}^{o} SQ_{jk}.F_j}{o}$$

The correlation table is crucial to customize these parameters since it represents the dependencies between the causes (M_j, P_j and F_j indexes) and the consequences (var_k parameters).

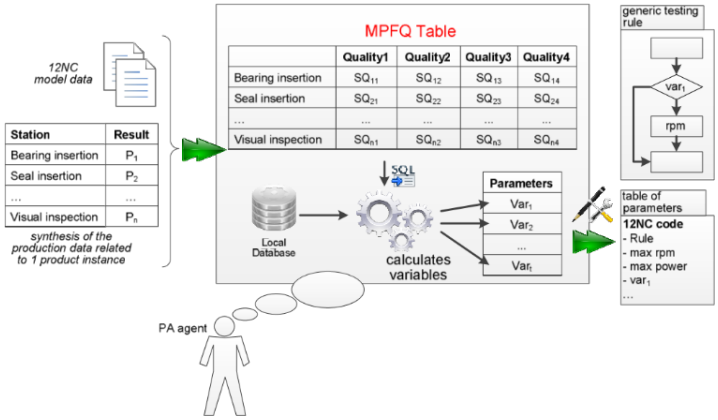

Figure 5.23: Customization of the parameters of the functional tests plan [165].

The adaptation function considers the data provided from all stations along the line, but particularly the data from the following stations: bearing insertion station, screwing stations, gap control station, visual washing unit inspection station and visual assembly inspection station.

As an example, considering the P_j collected along the production line reflecting the execution of the processes, the value of the var1 can be used to customize the testing rule represented in Fig. 5.23, namely:

- If the var_1 is set in the interval [0, 0.3[, no message is displayed to the operator;
- If the var_1 is set in the interval [0.3, 0.7[, the message nº 101 is displayed to the operator, warning to take special attention during the execution of the test;
- If the var_1 is set in the interval [0.7, 1], the message nº 102 is displayed to the operator, warning that something suspicious happened during the production which requires the implementation of additional procedures during the execution of the test.

The calculated var_k parameters are written in the "*Table of Parameters*", which posteriorly are used by the rule applied for the machine model of the appliance running in the PLC. When matching the generic testing rule and the var_k parameters written in the "*Table of Parameters*", the functional test plan is customized for the particularities of the produced washing machine.

5.3.4.5 Adjustment of the Correlation Parameters

Complementary to the adaptation of the functional test plans performed by the PA, IMA may embed strong global adaptation mechanisms, namely to support the modification of the rules, the triggering values of the testing rules and the adjustment of the SQ_{jk} values of the MPFQ correlation table. In fact, IMA does not operate individually, but instead collect and combine the decisions/information of individual agents, providing warnings and suggestions to adapt the system to specific events that may occur in the production environment. Particularly in this work, IMA modifies the policies/rules of the correlation functions associated with the functional tests plan. Thus, IMA may use

statistical correlation analysis to verify the correlation of SQ_{jk} values to the "real" influence of the related materials, processes or functions on a specific quality feature. Hence, SQ_{jk} parameters may be tuned, *e.g.* especially in the start-up of new products, establishing self-learning behaviour

5.3.5 Scenario 2 - Self-adaptation and Self-optimization Mechanisms

In this section, the local adaption will be focused on the Vision Inspection Stations. Each Vision Inspection Station performs a set of different checks. Particular for the Washing Unit (WU): position of the belt on the motor, **belt thickness**, the correct mounting of the heater, the presence of the clamp blocking the exhaust pipe. The focus on this section is the belt thickness inspection.

These tests are mandatory to pass. Otherwise, the QCA agent will recognize this failure, and it will be shown a failure notification to trigger a manual repairing process. Interestingly, in this project one of the partners (i.e., Università Politecnica delle Marche) has used the principles of self-* on robot vision prototype with success, as illustrated in Fig. 5.24.

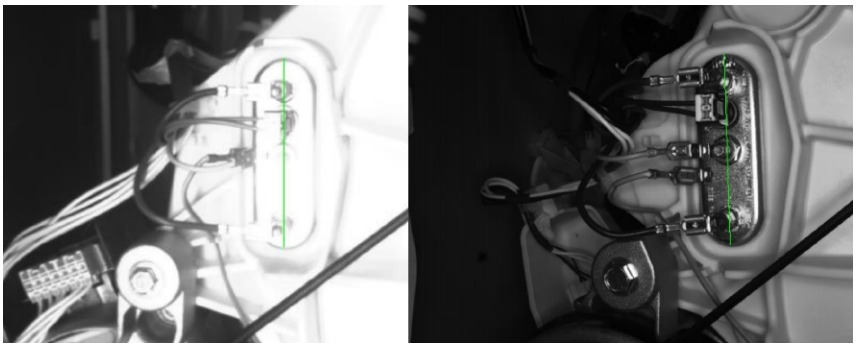

Figure 5.24: Image without adaptation, and on Right-side image with software and hardware self-*. The "Hardware" self-X strategies (e.g., Camera auto exposure driven by the evaluation of the image quality) and ii) "Software" self-* strategies (e.g., Automatic contrast recovery) [213].

As illustrated the Fig. 5.24, refers to a washing unit model that is closer to the existing illuminator, thanks to the self-*, the camera was able to keep the image quality to the desired level for that washing machine's model [213].

Next, the inspection process related to the position of the belt (and its thickness) guarantees good transmission of the movement from the motor to the pulley that rotates the drum. The measured values of the size and the position of the belt are used to calculate the Performance Index related to the vision station. Along with the identification of the washing machine's model, and the QCA agent sends information regarding the tests to the QCS Station. The self-adaptations are executed by the QCA, since it is the agent that knows how to react and when to adapt to the different models of washing machines and to whom (i.e., QCA). The combination between the QCA and the QCS (see Fig. 5.25) exhibit some CPS principles, the intelligent part (the agent) is related to the the physical

part (the hardware responsible for the inspection tasks), which in this work also embodies a measurement system developed in LabView™ [194].

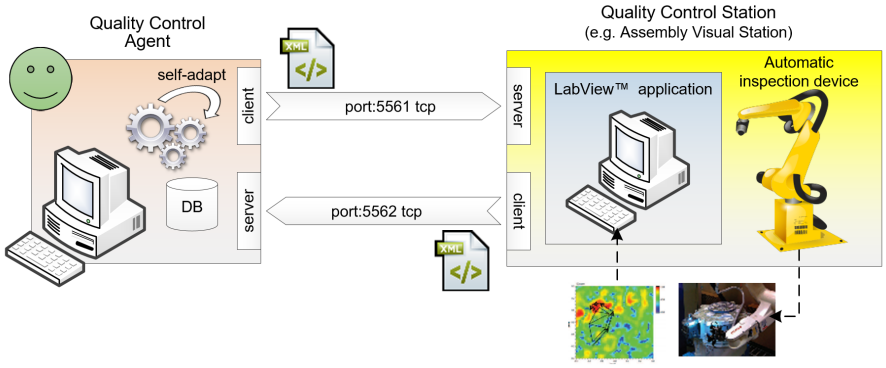

Figure 5.25: Interconnection of QCA and QCS by the TCP/IP sockets to exchange information related to quality control tests [112]. In detail, QCA (left-side) invoke a service inspection test, which triggers an automatic inspection device for quality control. Finally, the LabView™ perform the measurements and reply with the to the QCA with the diagnosis result.

The Data exchange between QCA and QCS relies on opening two TCP/IP sockets: one for the communication from the QCA to the QCS and the other for the communication from the QCS to the QCA. The exchanged messages are described in XML to simplify the common understanding. For this purpose, two XML templates are defined (see Fig. 5.26): one to send the data from QCA to the QCS and another one to exchange data from QCS and QCA. Briefly, the QCA provides the instructions for the test plan to the QCS containing mainly the sequence of tests to be performed, the specifications of the components assembled in the product and the parameters for the physical equipment and the inspection algorithms, together with the cycle time that is the maximum allowed time to perform the operations.

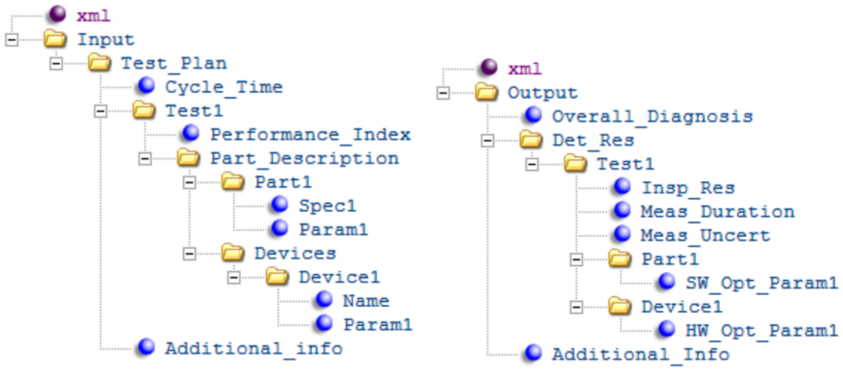

Figure 5.26: Extract the XML structure used to exchange data from the QCA and the QCS. Containing the instructions structure of the inspection sent by the QCA to the QCS (left-side), structure of the result of the inspection tests provided by the QCS to the QCA (right-side).

After the measurement procedure is computed, it is necessary to give this information to the QCA agent calculate the P_j and send it to the Product Agent to feed the MPFQ model.QCA receives the results comprising a global score that represents an overall result of all the performed tests and the detailed results of each test that are composed of the

quality score, the measurement duration, and its uncertainty.

If the P_j in this process is continually "Perfect", this means that some Functional Tests at the end might be skipped allowing to save time, on another hand if this QCA is indicating lousy quality, an intervention might be scheduled. Next, are represented the measured values of the size and the position of the belt are used to calculate the P_j related to the vision station:

$$P_{VIS} = \sqrt{P_s P_p}$$

$$P_s = e^{-\frac{(V_s - X_s)^2}{2(2\sigma_s)^2}}$$

$$P_p = e^{-\frac{(V_p - X_p)^2}{2(2\sigma_p)^2}}$$

Where:

- V_s is the size of the belt measured for the current product under test;
- V_p is the position of the belt measured for the current product under test;
- X_s, X_p are the mean values calculated on the products of the same model that have been already tested and are their standard deviations.

A low P_j value reveals a misalignment of the belt on the pulley as shown in the Fig. 5.27.

Figure 5.27: Examples of acquired images and corresponding Performance Index [213].

The result of the calculation of the P_j of the different models is saved by the RA and sent to the PA. The RA keeps a graph with the P_j calculation along the time per washing machine model. An important phase is the thresholds that are calculates and send to the QCA, which are necessary to compute the P_j. As an example, a in (Fig. 5.28) represents the P_j value over time of the several type of models. In the application of the algorithm to the belt position acceptance thresholds when a new model is produced.

$$T_p^{max} = \mu_p + \eta\sigma_p$$

$$T_p^{min} = \mu_p - \eta\sigma_p$$

where:

- μ_p, σ_p are the statistical model values of V_p (measured belt position value);
- σ is a coefficient.

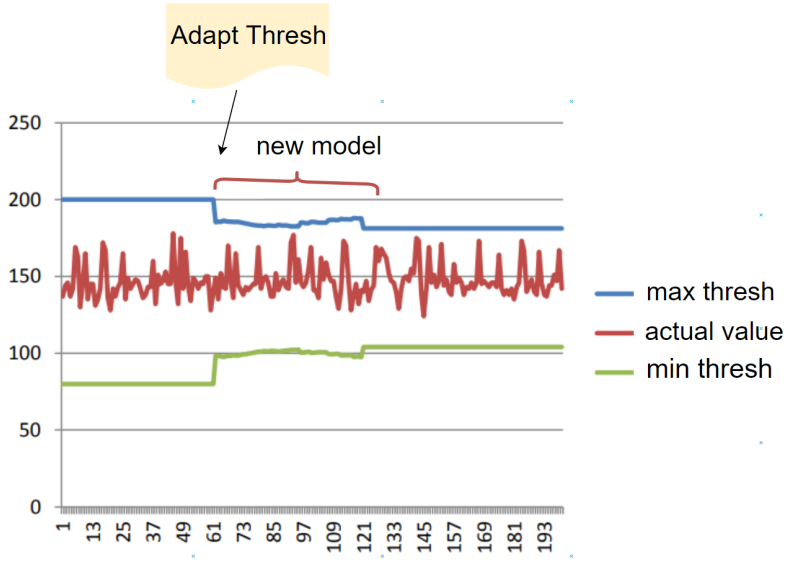

Figure 5.28: Example of a Belt Position Acceptance Thresholds Adjustment (pixels), when a new model is produced [152, 213].

These thresholds are progressively updated until the number of inspected items is higher than a defined number (e.g. 100). When this number is reached the thresholds of the model are fixed (value is reduced; e.g. from 5 to 4). For a deep understanding see [213].

When a new model is produced the default acceptance thresholds are applied may need operator expertise to manually set the thresholds if he already knows them. After a defined number of items (*e.g.*, 50) the statistical model has enough information to compute the thresholds. This means that the QCA receives the results global analysis result of all the performed tests and the detailed results of each test that are composed of the quality score, the measurement duration, and its uncertainty. The measured values of the size and the position of the belt are used to calculate the Performance Index related to the vision station:

This allows to detect deviations in advance and to trigger the implementation of corrective measures (*e.g.*, the automatic calibration of the tool wear or the light source, or to request external maintenance intervention) to mitigate the detected deviation.

5.3.5.1 Adaptation Rules

Adaptation rules in a QCA based on the WtR strategy can be used to adapt the optimal parameters of a service, in this case a test plan. When a new model of washing machines arrives in the production line, or to adapt the parameters of an already existing test

plan in order to deal with drifts and local variations of the optimal solution [194]. Each quality control task implements both kinds of adaptation rules, following the methodology introduced in [16].
In the following Section, one specific example, namely the **Vision Control Station** is addressed and the adaptation strategies of some of the test plan parameters are shown.

5.3.5.2 Vision Control Station

The Vision Control Station that is located in the first part of the Washing Unit line. The Vision QCS sends to the Vision QCA the following information (in addition to other):

- **WU1_RES:** result of the test (OK/KO);
- **WU1_BELT_SIZE:** value of the belt size measured by the QCS;
- **WU1_BELT_POSITION:** value of the belt position referred to the shaft measured by the QCS.

The Vision QCS receives from the Vision QCA the following information (in addition to other) in order to adapt the criteria for the selection of good and faulty items:

- **WU1_BELT_MIN_THRESH:** min. value allowed for the belt size measurements;
- **WU1_BELT_MAX_THRESH:** max. value allowed for the belt size measurements;
- **WU1_BELT_POS_MIN_THRESH:** min. value allowed for the belt position measurements;
- **WU1_BELT_POS_MAX_THRESH:** max. value allowed for the belt position measurements.

These parameters are strongly depend on the model of the items produced and their value must be optimized. Let us suppose that a new model of washing machine arrives, and the test plan is not available. The Vision QCA uses the following rules in order to find the optimal parameters for the new test plan:

- The belt thickness WU1_BELT_SIZE (Th_i) and belt position WU1_BELT_POSITION (BP_i) are calculated from the item i and stored in the QCA local database;
- Then the estimates of the optimal test plan parameters are updated as follows:

  Th_i is used to update WU1_BELT_MIN_THRESH (Th_l) and WU1_BELT_MAX_THRESH (Th_h) with the ± 3 standard deviation interval around the mean value Th; values outside the ± 4 standard deviation interval are discarded as outliers;

  BP_i is used to update WU1_BELT_POS_MIN_THRESH (BP_l) and WU1_BELT_POS_MAX_THRESH (BP_h) with the ± 3 standard deviation interval around the mean value BP; values outside the ± 4 standard deviation interval are discarded as outliers.
- From the M iteration on, a Confidence Level index (Cl) is calculated on the estimates

of both (Th_l, Th_h) and (BP_l, BP_h) using the following equation:

$$Cl = (1 - \sigma/X) * 100$$

where σ is the standard deviation and X the average of the specific parameter both calculated over the last M iteration values. When the confidence level exceeds a predefined threshold ($Cl \geq Cl_opt$), or in the case a skilled operator approves the estimated test plan parameters, the temporary test plan becomes an effective test plan.

The parameter M is an integer value, Cl_opt is a percentage and they can be configured by the Quality Manager.

5.3.6 Deployment in The Factory Plant

This sub-section describes the implementation, installation, and operation of the MAS solution in the factory plant.

5.3.6.1 Implementation of the MAS Application

The application was implemented using the JADE framework, which provides a set of functionalities, such as white and yellow pages services [15], which simplifies the development of agent-based applications, as well as a set of auxiliary agents, *e.g.*, agent communication channel (ACC) and agent management system (AMS), which support the life-cycle management of such applications.

- Each type of agent, namely, PTA, PA, MA, and IMA, and IMA, is a Java class that extends the agent class provided by the JADE framework according to the specific behaviour of the agent (modelled and validated using the Petri nets formalism). The agents were developed following the JADE's behaviour concept [15] that allows the execution of several actions concurrently;
- The communication among the distributed agents is asynchronous, *i.e.*, the agent that sends a message continues its execution without the need to wait for the response. To achieve standardized communication, the exchanged messages were encoded by using the FIPA-Agent (ACL) and their content formatted according to the FIPA Semantic Language (SL0) [68].

5.3.6.2 Installation of the MAS Application

The agents deployed for the Industrial factory plant were as follows.

- One IMA;
- Eleven QCAs, namely, for the gap control, washing unit inspection, assembly vision check, vibration, final assembly visual check, and six boxes of FT workstations;
- Six RAs, namely, for the A-bearing, B-bearing, pulley screwing, screwing upper counterweight, screwing front counterweight, and on-board controller workstations;
- Several PTAs, namely, one agent for each washing machine model.

These agents were distributed by eight computers (mainly, Core 2 Duo, 1.66 GHz, and 1 GB RAM) disposed along the production line [114], as illustrated in Fig. 5.29.

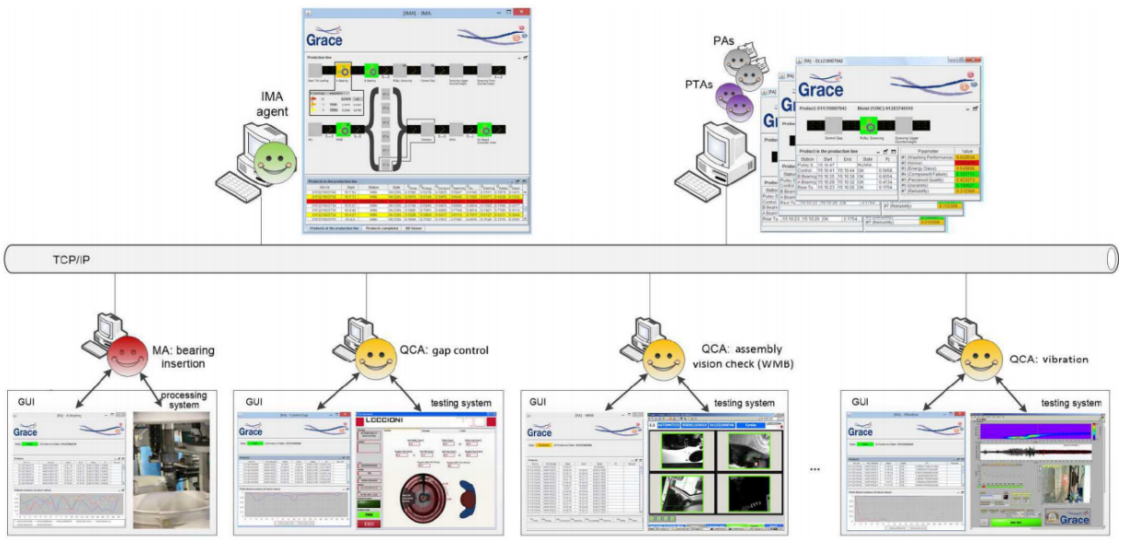

Figure 5.29: Distribution of agents in the installed system at the Industrial production line (source [114]).

The interconnection among the agents was performed by using TCP/IP over an Ethernet network, as well as the interconnection between the agents and the PLCs located at each workstation. Agents were customized according to their particularities, namely, the type, list of skills, and set of algorithms, by interpreting an extensible mark-up language (XML) configuration file during the start-up phase.

PAs are launched on-the-fly by the associated PTAs according to the batch of orders provided by the MES system of the factory plant. In a stable production flow, approximately 400 PAs are running simultaneously.

5.3.6.3 MAS Solution Running in Practice

The agent-based solution installed in the Industrial production system was intensively tested, being possible to analyse its functions related to monitoring, data analysis, and **self-adaptation** mechanisms, performed by the agents at **local** and **global levels** [152]. As example, RAs are continuously analysing the performance of their workstations to detect any degradation in their behaviour. This allows to anticipate deviations and implement proper self-corrective measures or request external maintenance interventions to mitigate these problems [165].

Global and Offline Reconfigure Mechanism

In the background, IMAs apply trend analysis mechanisms data – mining techniques to generate potential configurations (i.e., self-optimization guidelines) to be sent to the DB

pool of candidate (re)configurations of the RA and PA, as illustrated in Fig. 5.17. Analysis of the evolution of the Q_i indexes for each product along the line, being compared with the average values for the same indexes for its model. This allows to detect if the quality of this particular product is being above or below the average standard values. Figure 5.30 illustrates the results for a specific washing machine.

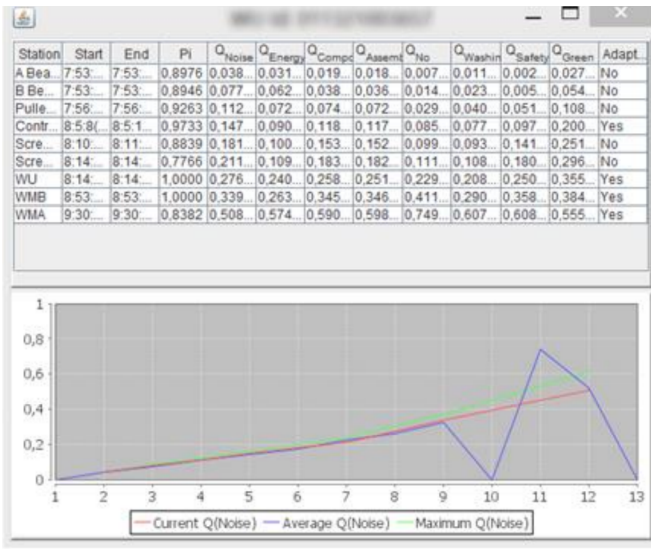

Figure 5.30: Evolution of the Qi indexes for each product along the production line.

Analysis of the Q_i indexes for each product and generation of warnings in any point of the production line (*i.e.* at any point of the production) in case the desired quality is not possible to be achieved anymore. Note that a yellow warning is generated if the achievement of the desired quality is in risk, and a red warning is generated when the system detects that it is not possible anymore to achieved the desired quality (even if the remaining operations will be performed with Pi equal to 1). In Fig. 5.31 it is possible to verify that some products are marked with yellow and red (in this case meaning that the desired quality will never be reached).

As an example, IMAs close the feedback loops by using the results of this data analysis to support the dynamic adjustment of up-stream production processes, *e.g.*, the adaptation of the screwing process stations.

Similar approaches based on data-driven techniques can be combined with this agent-based infrastructure, other data analysis and self-adaptation procedures were implemented and successfully operated at the production line. Particularly, the analysis of the evolution of the quality indexes at station level and the online adaptation of the process plans for each washing machine model, performed by PTAs using the feedback information from PAs.

Using the MPFQ approach, it is possible to correlate the information related to materials, processes, and functions to obtain an estimation of the quality of the washing machine.

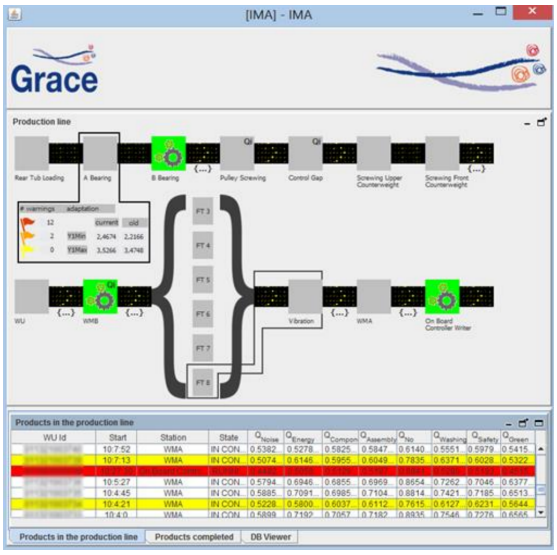

Figure 5.31: Generation of warnings based on the analysis of the Qi indexes for each product.

Based on the WtR trend mechanism, PAs can continuously analyzing the evolution of these quality indexes, which are related to noise, energy-saving, component conformity, assembly conformity, no-leakage, washing performance, safety, and green footprint.

The dynamic correlation of this data allows generating warnings, in any point of the production line, in case the desired quality cannot be achieved anymore, even if the remaining processes will be carried out with a performance of 100%. This mechanism allows identifying earlier possible quality deviations and based on the HtR strategy defined react with different type of reconfiguration actions deciding to remove the washing machine from the production line to save time (*i.e.*, removing earlier a washing machine that will not be quality conformant) money (*i.e.*, reducing the scraps resulting from producing nonconformant products).

5.3.7 Main Results

The Industrial Cyber-Physical Production System deployed in the real Industrial production line was intensively tested during its operation, is possible to extract some quantitative and qualitative benefits [110, 151].

5.3.7.1 Qualitative Properties

The experimental tests showed that the installed agent-based system reaches several important qualitative properties:

- ***Modularity and flexibility:*** the use of distributed software agents simplifies the development of complex computational software applications by dividing the complex problem into simple ones. This allows achieving modularity since the system specifications are built for 4 types of agents, *i.e.*, PTA, PA, RA and IMA, each one exhibiting proper behaviour. For the installation in the Naples factory, several instances for each

type of agent were created, each one using the same agent codification and inheriting its behaviour but customized for its particularities according to a XML file;

- ***Distribution:*** due to the distributed architecture provided by the MAS system, the distribution of large-scale systems is easier as agents might be distributed by hardware computational resources according to the application needs, *e.g.*, geographical dependency or processing capabilities. In the installed system, the agents were distributed by several PCs disposed along the production line;
- ***Runtime and on-the-fly-reconfiguration:*** in such systems, agents can be removed, others can be added or even some modifications can be performed in the behaviour of the agents without the need to stop, reprogram and reinitialize the other components (*i.e.*, the system can continue running without any perturbation). In the installed system, this is illustrated in several ways, e.g. shutting down RA (*e.g.*, the one associated to the A-Bearing station), adding new RA during operation (*e.g.*, associated to functional tests boxes) or, changing the algorithms embedded in the RA associated to the Screwing Upper Counterweight station. These functionalities were successfully tested and validated (note that in a centralized implementation, this feature is not possible);
- ***Robustness:*** in case of an individual node breakdown, the system continues running without perturbation (in opposite to the traditional centralized structures where the system collapses). In the installed system, this is illustrated by shutting down some RAs, *e.g.*, those related to the functional tests stations, or IMA, without affecting the global behaviour of the system;
- ***Adaptation:*** the use of distributed control structures allows the run-time adaptation, *i.e.*, applying local self-adaptive concepts to adapt the system behaviour according to unplanned changes. In the installed system, this feature was illustrated by the adaptation of the parameters to execute the processing/inspection operations, to customize the sequence plan of the functional tests and to customize the flow rate parameter of the on-board controller of each washing machine. Also observed in the installed system is the fast adaptation of the system to the change of washing machines models in the production sequence, and also to the introduction of new product models in the production line;
- ***Scalability:*** a common limitation of MAS solutions is usually associated to the agent middleware (in this case JADE) and it is related to possible delays or congestion in the communication infrastructure due to the growth of the exchanged messages (note that the increase of agents implies an increase of exchanged messages, not in a linear way, but a more exponential manner). From the achieved experimental results, the increase from several agents to approximately 400 agents running simultaneously and having physical representation, namely connected to washing machines and workstations, some of them exhibiting weighted GUIs, did not provoke visible degradation at this level;
- ***Smooth migration:*** the use of MAS technology allows the smooth migration from

old technologies/systems to new ones. This is illustrated in the installed system with the consideration of 12 workstations, from the entire production line, to be controlled by the MAS solution; a slowly and smoothly integration of the remaining workstations along the production line can be performed gradually in the time.

5.3.7.2 Quantitative Impact

The installed agent-based system brings some significant improvements in the factory plant, namely in terms of:

- **Increase Production efficiency**, *e.g.*, due to the adaptation of the process parameters, the adaptation of the functional tests for each washing machine and the customization of the onboard controller parameters, based on the production history. As an example, with this approach, the functional test time for each washing machine is reduced by approximately 20%, which implies an increase of the flow in the production line or in alternative the possibility to save one testing box (and consequently save investment and energy);
- **Reduction of the production downtimes:** The appliance industry is facing relevant costs for down-time due to the frequent change of models. The self-adaptation of the process parameters permits to reduce the downtime of workstations, by approximately 10%, when new models are produced. Using this human intervention approach is only required for 20% of new models;
- **Reduction of nonconformities and an increase in product quality:** The costs of non-quality (*e.g.*, production inefficiency cost and service cost) are also essential to be minimized in this type of industry. The introduction of more effective and self-adapted quality control procedures allows to increase the product quality and to reduce the nonconformities of about 1,5%;
- **Increase in the product customization:** Each washing machine is customized, and its operation optimized due to the adjustment of the parameters of the on-board controller based on the historical production data. This allows achieving better product quality and customer satisfaction;
- **Cost reduction of the scrap costs**, since the earlier identification of defects or quality unconformities in the washing machines being produced can lead to a reduction of the scraps from 5% to 3%. Additionally, stopping, in advance, the production of washing machines that never reach the desired quality contributes to reduce the waste costs and also save production time;
- **Improvement of product energy efficiency:** The customization of the parameters of the on-board controller has a significant impact on the sustainability of the domestic appliance use. For example, the calibration of the control flow valve implies that more precise washing machine programs are installed, leading to a reduction of 5% in the water and energy consumption;
- **Product quality**, since most effective quality control procedures are performed along the production line, *e.g.*, using customized testing plans in the functional tests area,

and since the on-board controller of each washing machine is customized according to the historical production data.

5.3.8 Discussion - Lessons Learned

The experience gathered from the development and installation of the agent-based solution in a real Industrial production line allowed to learn some important lessons. These lessons are intended to mitigate future risks based on inform process improvements and promote best practices for the future.

The first one is related to the design-deployment path of multi-agent system. The design, debugging and deployment of agent-based systems are a complex process, usually performed in an ad-hoc manner. Traditionally, the correctness of the design can only be validated after the implementation phase, leading to a very time-consuming design-implementation process that presents high rates of misunderstanding and mistakes, and, as a consequence, it is very expensive [30]. The use of a formal methodology, based on the Petri nets, allowed a rigorous specification of the proposed solution, and posterior validation and simulation during the design phase, ensuring that the model represents correctly the specifications of the real system, permitting the correction of errors and misunderstandings, and improving the control strategies before the implementation phase.

In these scenarios, the offline tests are important to ensure that errors are corrected, but they do not replace the use of online experimentation in the Industrial production line, since there are situations that arise only in real Industrial environments. For this reason, it is important to properly balance the negative impacts of occupying the production line for testing and the benefits of deploying the system in a real environment.

The second lesson is related to the use of agent development frameworks. In fact, the use of an agent development framework simplifies the development of MAS solutions, since it provides a set of important functionalities that support the development, debug and operation of these systems. However, these frameworks, present some drawbacks, namely the existence of a centralized node reflecting the DF (Directory Facilitator) service, the system scalability and the performance affected by the use of Java synchronized methods [205], which should be improved in the future.

A particular problem is the lack of compliance when deploying MAS systems to Industrial environments, which is mainly a problem of standardization in the field. FIPA, which is the main standardization body in the field, made a very important job in defining the specifications to develop MAS systems, but important issues are currently missing and require further standardization work. Examples are protocols that better fits the behaviour of Industrial control systems, event notification at low control level, and integration with legacy systems in a transparent manner.

The installation in a real factory plant confirmed that real Industrial production environments are different playgrounds from those found in theoretical, simulated and laboratorial ones, presenting very challenging and demanding requirements and technical constraints. As an example, Industrial environments exhibit strong constraints in terms of communication infrastructure that impose additional technical problems to the implementation of MAS solutions.

Interestingly, the purpose of use services is to help reduce the integration effort, but the integration with legacy systems, *e.g.*, LabView™ applications and processing stations, are very time consuming and usually developed one-of-a-kind according to the particularities of the hardware/software devices, constituting an important task that increases the complexity of the deployment process. Additionally, the equipment disposed in such Industrial environments has usually close and proprietary protocols being difficult to develop the required interfaces.

Finally, the successful installation of the agent-based solution on Industrial Line production machines contributes to reducing the industry's skepticism of adopting MAS technology in the short term and to addressing the emerging problems they are facing.

5.3.9 Limitations

Some of the limitations were already addressed (see 5.3.8), but the recent technological progress into data analysis is worth mentioning. One of the fastest-growing transformations in the Industry 4.0 trend is the ability to more accuaretly predict the future, specially if this computational power runs in the Cloud technology, where the level of responsiveness is expected to increase. Given this scenario, the *offline reconfiguration* was considered in the development of ADvISER agents, to enhance the behaviour of the system with advanced data analysis algorithms and simulation tools. However the system was not fully explored to consider large amounts of data, which could be acquired from all types of sensors in advanced manufacturing. In some cases this type of extensive data analytics is beneficial, as an example Zero Defect Management for both preventive and corrective maintenance.

In both case studies, the local *WtR* prediction models, only consider small data volumes collected in real-time, instead of using complicated models that required huge training datasets. Consequently, these predictions must be carefully analyzed before any reaction. In a context where it is difficult to have enough training data, "Transfer Learning" techniques are useful to overcome the cold-start problems. Nevertheless, the flexibility of ADvISER, due to be service compliant, allows overcoming potential limitations, transforming them into opportunities in order to welcome the most recent tools developed for smart manufacturing.

5.3.10 Evolution of Automation System Architectures

In the implementation of the case study it was possible to put into practice the some of the challenges mentioned by Foehr et al. in engineering of next-generation Cyber-Physical Automation System Architectures [67], in particular the evolvability based on the flexibility and reconfigurability that CPS offers, this means that it was possible to migrate, to a limited extent, to a new generation of distributed automation system [114], without down-times, as identified in [83].

The lower control level using PLCs running IEC61313-3 programs was preserved to ensure the real-time control and the MAS solution was placed at the higher control level to introduce intelligence and adaptation to the system performance. The experts agree [67] on the fact that the smooth migration from traditional automation systems into the new generation of distributed automation systems are crucial since legacy systems will continue running and will co-exist with the new systems.

The reconfiguration of the computational system can be easily achieved in runtime, since agents can be removed, added, or even modified on-the-fly, *i.e.*, without the need to stop, reprogram, and reinitialize the other agents. This important characteristic, which is not possible in the traditional implementation, as illustrated in several ways, *e.g.*, shutting down an RA associated to the seal insertion station, adding a new RA related to a new box of the functional tests station or changing the algorithm to calculate the performance index embedded in the RA associated to the bearing insertion station. The integration of process and quality control allowed to implement runtime self-adaptation procedures, embedded in local and global agents, providing fast responsiveness to condition changes in the production process. Examples of such adaptation mechanisms are the customization of the functional tests plan and the on-board controller of each washing machine [114].

Another important characteristic, observed during the operation of the agent-based system installed in the factory plant, is the possibility of ADvISER vision of **evolvable**, in the sense that it was possible to execute a smooth migration from the existing technologies or systems to the adoption of new ones. As an example, the installed solution considered 12 workstations to be controlled by the agent-based solution, but a slowly and smoothly integration of the remaining workstations along the production line can be performed gradually in time. In the end, and probably one of the major contributions of this work, is the assessment of the MAS benefits in real Industrial environments, by demonstrating its effective applicability in a real Industrial factory plant.

5.3.11 TRL Analysis

This case study is closer to the maturity required by the industry (*i.e.*, TRL 5 - 7). The TRL levels related to this case study are as follow:

- Development of a MAS architecture for integrating process and quality control, with

a classification of TRL-5;
- The installation of the MAS infrastructure for integrating process and quality control, classified as TRL-6;
- Feedback control loops for online process parameter adjustments are classified as TRL-4;
- The WtR strategies developed for ADvISER can be addressed separately:
  - The Adaptation mechanisms for the functional testing plans and product customization are classified as TRL-4;
  - Self-adaptive quality control systems are classified as TRL-5;
- The approach for the Distributed data collection was classified as TRL-5.

With these technology readiness levels described above, both the ADvISER's architecture and mechanisms are placed in a good TRL level, which is important inside the I4.0 Industry vision.

5.4 Summary

Here two case studies were developed to test the proposed ADvISER approach in real case scenarios and demonstrate its effectiveness in engineering self-reconfigurable software manufacturing systems.

It is important to mention that the first common challenge to validate these types of systems is the lack of benchmark opportunities to test intelligent manufacturing technologies. The process of selecting case studies and the need of implementing platforms to support ADvISER implementation, naturally took some time. But this was a fruitful process that allowed the application of the developed multi-agent architecture in two real-world cases. Together they not only demonstrate ADvISER's flexibility, in the different TRL phases tested, as they demonstrate their generic profile. Moreover, as indicated in the defining motivation of CPS, the ability to introduce agents collaborating with services is essential, if we are to pursue the trend of system development and computer embodiment within smart manufacturing. As emphasised throughout the ADvISER design it is necessary to associate software agents to specific stations acting on the line and performing process operations. This process implies that the hardware is associated with each agent, and the device procedures can be accessed through a service interface. Based on this digital versus physical combination it can be derived that ADvISER fits the needs of an Industrial Cyber-Physical Production System.

Recalling the ADvISER initial requirements that need to be observed (refer to Section 2.5) aiming to execute a truly dynamic, intelligent and pro-active service reconfiguration, the achievements obtained in the AIP-PRIMECA and Whirlpool Use Case can be summarized as follows:

- ADvISER responsiveness **(IR 3.1)**, was demonstrated in the adaptation rules in the

Whirlpool use case when adapting the service parameters of the inspection tool and on the responsiveness product change-over (*i.e.*, the change of the product models in the production sequence). For the AIP-PRIMECA case study, the responsiveness can be observed on the reaction to service disturbances;

- The proactiveness **(IR 3.2)** was demonstrated in both cases. In the first case study, the proactiveness can be verified on the offline mechanism which is responsible for proposing (re)configurations. In the second case study it can be observed when preventing the generation of defects and their propagation, and on the dynamic adaptation of the functional tests plan;
- The Learning ability **(IR 5)** was demonstrated on both case studies: the reinforcement learning in the AIP-PRIMECA case study and in the Whirlpool case study on the adjustment of the parameters of the on-board controller;
- The cooperative and collaborative capacity **(IR 6)** is one of the pillars of the distributed agent-based system. However the collaborative requirement can be better demonstrated in the AIP-PRIMECA case study;
- The capacity to Interact with legacy systems and physical devices **(IR 7)** is also present in both case studies by means of using Standards (*e.g.*, FIPA-ACL), but mostly due to the use of Services (*e.g.*, ILOG, R-Server, LabView™ applications);
- The Smooth reconfiguration ability can be seen as the gradual process of improvement in runtime by the responsiveness **(IR 3.1)** and pro-activeness **(IR 3.2)** (mostly performed by small and logical self-adaptions) for change without breaking the system architecture.

The implementation of the distributed multi-agent system architecture into two different case studies (each one providing distinct requirements) increases the chances of having the agent technology adopted by the Industry. Together they not only demonstrate ADvISER flexibility, in the different TRL phases tested, as they demonstrate their generic profile.

In the next chapter, we summarize the contributions made throughout this work where it is possible to answer the initial research questions. Some additional research issues will be addressed in the "future work" section.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

"Discovery consists of seeing what everybody has seen and thinking what nobody has thought"

Albert von Szent-Gyorgyi

This chapter overviews the dissertation and presents the final remarks and list of achieved contributions. It also offers a critical look over the developed work, by reviewing the state of the art and discussing how the underlying research area(s) was (were) pushed forward with the work reported in this document. Finally, the research challenges left open, or opened meanwhile, are discussed and presented as future work.

6.1 Introduction

Starting from the last decade, the smart manufacturing sector has been witnessing the rise of a fourth industrial and technological revolution called Industry 4.0, empowered with an unprecedented capacity to enhance smart manufacturing automation processes with reconfiguration mindset.

To support this evolution, production systems should evolve to become more configurable and flexible for the higher number of parameters as possible, in order to maximize the production performance. This puts reconfiguration at the heart of any agile and smart manufacturing system. Consequently, researchers' interest in new IT approaches, capable of facilitating changes in the structure and behaviour of the system beyond the traditional approaches, has been triggered. In this scenario, automatic and intelligent models are necessary assets. The study of service reconfiguration into the domain of Industrial automation, as well as other like intelligence, promised to deliver an efficient way to manages and adjusts the control system.

Unfortunately, building manufacturing applications capable of being reconfigured is an extremely complex and challenging task, but crucial for a profound long-term impact. This work aimed at delivering an architecture that facilitates the self-adaptive and self-configuring manufacturing operation.

6.2 Main Contributions - Confirmation of the Hypotheses

With ICT's advances, it is now possible to adopt distributed and adaptive systems in an easier way than it was a few years ago. As a result, many industrial manufacturing systems contain characteristics that allow the implementation of adaptable systems with reconfiguration capacity (either to adapt or to evolve) at the speed of the new requirements.

In this activity, the dynamic service reconfiguration is an essential technological area that allows designing adaptable adaptable and reconfigurable systems on-the-fly, *i.e.*, without the need to stop, reprogram and restart the system. Due to the interest in redesigning systems capable of supporting evolution, such as the analysis of a large number of distributed information, or the rapid need for decision-making, the dynamic service reconfiguration process has become a hot topic in smart manufacturing systems, particularly with the Industry 4.0 initiative.

In this work, the features and requisites of how such systems are used in manufacturing were reviewed. Research literature in the field showed that service reconfiguration is usually performed in a manual, offline and centralized manner, without fulfilling the requirements for truly automatic service reconfiguration.

The proposed ADvISER approach provides core functionalities to execute a dynamic, online, and decentralized service reconfiguration. The service reconfiguration process itself consists of a software solution based on intelligent software agents empowered with service technologies, which work together with the control application that is under reconfiguration, (*e.g.*, AIP-PRIMECA and Whirlpool use case). This enables to:

- Apply different strategies to identify reconfiguration opportunities, *e.g.*, while facing service failures, service performance degradation or production changeover, in a pro-active manner;
- Optimize the manufacturing operations collaboratively.

The proposed service reconfiguration solution was implemented by using the agent-oriented JADE framework and tested in a flexible manufacturing system use case. The experimental results confirm the feasibility of the proposed approach, with the services reconfiguration procedure used displaying improved system performance than the normal operation, and the use of the collaborative mechanism leading to even better results in collaborative situations.

In this section, the research questions presented throughout the document will be recovered:

> **RQ1: What are the main engineering design process guidelines and principles that should be considered in reconfigurable systems, particularly to support an effective decision-making process for a robust reconfiguration evolution?**

The answer behind this question is an agglomeration of several vital aspects that systems must exhibit to support reconfigurability and evolution. The guidelines identify key processes to be able to apply the reconfiguration principles, namely *why*, *what*, *when*, and *how* to proceed with a reconfiguration procedure. In Chapter 3, we looked through the *8-features*, which gave some insights about how these flexible and reconfigurable systems can be designed, based upon the *7 industrial requirements*. Naturally, this question led to the creation of two more sub-questions:

i) In which cases is the reconfiguration likely to be implemented? The answer acknowledges two distinct situations. Firstly, adaptation to guarantee the performance of the service or its initial purpose (*e.g.*, failures or quality loss). Secondly the evolution is guaranteed with the focus on the learning mechanism, the WtR methods developed in Chapter 4 allow an understanding of this sub-question.
ii) *What are the options to detect the need for reconfiguration, and how can such needs be mitigated?* The WtR model addresses this point through several mechanisms with different reconfiguration activation policies. The second part of the question focuses on mitigation mechanisms. For this purpose, in Chapter 4, an HtR model that considers the various reconfiguration actions was developed, improving the chances of mitigating possible problems.

> **RQ 2: How to engineer a flexible and reconfigurable architecture for exploiting adaptable algorithms capable of identifying reconfiguration opportunities at key moments and to provide practical step-by-step reconfiguration processes?**

In this thesis, it was verified that the combination of services and agent technologies offers an alternative approach that simplifies and improves the reconfiguration process. It was found that the proposed architecture, based on intelligent software agents, can dynamically and autonomously identify reconfiguration opportunities and execute the best reconfiguration procedures, improving the intervention effectiveness. Last, but certainly not least, these intelligent agents evolve after identifying opportunities.

The agents compute all the feasible alternative ways of reconfiguring their catalogue of services and decide for the most promising one.

The individual services reconfiguration approach is directly mapped in competitive

environments, where the selfish behaviour of the agents leads to a truly dynamic and decentralized service reconfiguration. In case of collaborative environments, a decentralized collaborative interaction protocol was designed to ensure that the intended reconfiguration, triggered by the individual agents, is only executed when it is beneficial for the whole system, avoiding reaching non-feasible configurations and promoting a more efficient system configuration.

6.3 Scientific Contributions

This section lists the most important and significant scientific contributions published during the development of this thesis. The summary of publications is the following:

- Conference Papers – 19
- Journal Articles (**Quartile Scores**[1] **of Q1**) – 3
- Book Chapters – 2

All the contributions are from international, IEEE sponsored peer reviewed conferences indexed, either by Scopus or Web of Science.

List of Journal Publications:

- N. Rodrigues, E. Oliveira, P. Leitão, **Decentralized and on-the-fly agent-based service reconfiguration in manufacturing systems**, Computers in Industry, 101 (2018), pp. 81–90.
- P. Leitão, N. Rodrigues, C. Turrin, A. Pagani, **Multiagent system integrating process and quality control in a factory producing laundry washing machines**, IEEE Transactions on Industrial Informatics, 11 (2015), pp. 879–886.
- P. Leitão, N. Rodrigues, J. Barbosa, C. Turrin, A. Pagani, **Intelligent products: The grace experience**, Control Engineering Practice, 42 (2015), pp. 95–105.

List of Book Chapters:

- P. Petrali, A. Pagani, F. Boschi, G. Tavola, P. Fantini, J. Barbosa, N. Rodriques, A. Ferreira, G. Angione "Use Case: White Goods" In Colombo et al. (Eds.). In Digitalized and Harmonized Industrial Production Systems: The PERFoRM Approach 2019.
- P. Leitão, N. Rodrigues, C. Turrin, A. Pagani, Multi-agent System for Integrating Quality and Process Control in a Home Appliance Production Line In Leitão, P., & Karnouskos, S. (Eds.). Industrial Agents: Emerging Applications of Software Agents in Industry. Morgan Kaufmann 2015.

List of Conference Publications (Indexed Scopus or Web of Science):

List of the most relevant publications made during this thesis. The complete list can be found in the documents attached to this thesis.

[1] according to the *Scimago Journal Ranking*

1. N. Rodrigues, P. Leitão, E. Oliveira, Dynamic Service Reconfiguration with multi-agent Systems, in Service Orientation in Holonic and Multi-Agent Manufacturing, Springer, Cham, 2017, pp. 307–318
2. N. Rodrigues, P. Leitão, E. Oliveira, Triggering strategies for automatic and online service reconfiguration, in Iberian Conference on Information Systems and Technologies, CISTI, vol. 2016-July, Gran Canaria, Spain, 2016.
3. N. Rodrigues, P. Leitão, and E. Oliveira, Adaptive Services Reconfiguration in Manufacturing Environments Using a Multi-agent System Approach, in Multiagent, In MATES 2015: Smart Things Working Together, Jörg P.Müller, Wolf Ketter, Gal Kaminka, Gerd Wagner, and Nils Bulling, Cottbus, Germany, 2015.
4. P. Leitão, N. Rodrigues, and J. Barbosa, What-if game simulation in agentbased strategic production planners, in IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, vol. 2015-Octob, 2015.
5. N. Rodrigues, P. Leitão, and E. Oliveira, Self-Interested Service-Oriented Agents based on Trust and QoS for Dynamic Reconfiguration, in Service Orientation in Holonic and Multi-Agent Manufacturing (SOHOMA'14), Nice, France, 2014.
6. N. Rodrigues, P. Leitão, and E. Oliveira, Dynamic Composition of Service Oriented Multi-agent System in Self-organized Environments, in IAT4SIS '14, Prague, Czech Republic, 2014.
7. N. Rodrigues, E. Oliveira, and P. Leitão, Self-organization Combining Incentives and Risk Management for a Dynamic Service-Oriented Multiagent System, in Technological Innovation for Collective Awareness.
8. N. Rodrigues, P. Leitão, M. Foehr, C. Turrin, A. Pagani, and R. Decesari, Adaptation of functional inspection test plan in a production line using a multiagent system, in Industrial Electronics (ISIE), 2013 IEEE International Symposium on, pp. 1–6, 2013.
9. N. Rodrigues, A. Pereira, and P. Leitão, Adaptive Multi-Agent System for a Washing Machine Production Line, in Industrial Applications of Holonic and Multi-Agent Systems SE - 19, vol. 8062, V. Marík, J. M. Lastra, and P. Skobelev, Eds. Springer Berlin Heidelberg, pp. 212–223, 2013.
10. P. Leitão, N. Rodrigues, C. Turrin , A. Pagani ,P. Petrali, GRACE Ontology Integrating Process and Quality Control, IEEE Industrial Electronics Society (IECON'12) Montréal, Canada, October 2012.
11. P. Leitão, N. Rodrigues, Multi-agent system for on-demand production integrating production and quality control, Holonic and Multi-Agent Systems for Manufacturing (HOLOMAS'11) (V. Marik, P. Vrba, and P. Leitao, eds.), Lecture Notes in Computer Science, vol. 6867, pp. 84-93, Springer Berlin / Heidelberg, 2011. doi: 10.1007/978-3-642-23181-0_8.

6.4 Industrial Relevance

The focus of this work is to control of the changes and deviations that happen in a manufacturing automation system that operates in a collaboratively way. To accomplish this,

ADvISER explores the concepts of Services and Intelligent Systems, concepts which have emerged in recent years, with well-established industrial-relevant technological solutions.

ADvISER aims to develop an accessible multi-agent system service-oriented platform, empowering anyone in the manufacturing of automation systems with the ability to develop and/or improve all kind of services that potentially improve the system's benefit.

Several literature contributions and observable changes in practice, across a variety of case studies, confirm the importance and applicability of approaches such as ADvISER in the field of industrial reconfiguration. The design of the reconfiguration mechanism, through the industrial cyber-physical system, can ideally be introduced in a traditional control system with minimally invasive solutions. This means having a positive impact with minimal interference in the traditional processes. However, this only happens with some types of changes, *e.g.*, if it does not involve manual changes of transition.

The decision support system can thus be expanded with the proposed approach without significant efforts. What is not always true, the integration of this solution into a technologically rigid system, where condition changes can be introduced with operational impact, can be a challenge, from a technological perspective and a mindset change. Due to the advancement of technology in the industrial domain, the conflict of a different generation is increasingly easier to resolve.

Finally, the proof of concept in two case studies with different TRLs demonstrates generalization characteristics to easily apply to a wide variety of manufacturing scenarios and domains.

6.5 Future Work

The work described in this document is not a closed cycle. After the implementation of the designed solutions in real cases, it will be necessary to improve other features that will naturally arise along with their implementation and use.

6.5.1 Improve the Reconfiguration Mechanism

Although the reconfiguration mechanism has been developed and its performance tested throughout this thesis, the same architecture can still be enhanced to be more efficient in situations where the entire environment is known from the outset.

As seen throughout this thesis, a common problem in dynamic environments is the uncertainty of several variables floating in unforeseen ways. This type of variability is always associated with some discomfort and stress due to the impacts it may have on the system. This raises serious concerns about the certainty of change. The solution to be explored in the future can be placed in systems with more data analysis so that a system

can be controlled and be able to understand the types of reconfiguration.

Balancing the number of reconfigurations that are taken with the uncertainty of not being optimal, is not trivial, notice that for each change the system behaves differently as its environment changes. Our proposal allows us to adapt in the face of improvements, but not reacting quickly or reacting with more knowledge can be beneficial in the long term. Therefore, in the future, it is necessary to overcome this uncertainty. To this end, it is suggested as future work exploring the mechanism to regulate how much "nervousness" should there be in the system, with the inputs from the data analysis, the idea is to avoid falling into chaotic situations and allowing to run the system smoothly when facing very dynamic environments.

6.5.2 Leveraging Machine Learning

One of the great revolutions of the last years in the area of manufacturing is given to ML and its compelling applications, causing manufacturers to turn to the ML to improve several processes. Throughout this work, some concepts have been used, but there are a plethora of algorithms that are proliferating in today's manufacturing, such as automating inventory optimization, through maintenance, operations, quality. It is often necessary the use of algorithms to leverage machine learning capabilities, and to increase processing. Training models that recognize patterns, or analysing a broader spectrum of information requires computational power. Otherwise, it may take days to come up with the right solution. As a consequence, the solution results in more hardware, which leads us to push processing to the cloud. Fortunately, most of the providers already have specialized and optimized ML services. Due to the many advantages, this type of study should be offered in decentralized manner.

6.5.3 Human

Many of the features presented are strongly aligned with the Industry vision, like adaptation and evolution along the time. However, there are some concerns about designing intelligent software in production systems. According to the vision of the CPPS, the cyber-physical components can optimize itself with minimal human intervention, leveraging the interconnectivity to collaborate with each other's components to reach a production goal. With this, a critical point is raised based on the situation of where to place the operator at the Industry 4.0 vision.

Throughout this work, the connection between human and machine has been demonstrated, but in a context for cognitive tasks (*e.g.*, possible reconfiguration validation) during the production phase and to perform manual tasks requiring manual services by the operator's part. The trend of designing manmachine systems is mainly focused on cognitive tasks, by taking full advantage of supporting technologies like augmented reality [35, 57]. For instance intelligent dashboards make the maintenance operator conscious of

the pre-diagnosis and suggest what intervention has to be planned with other supporting technologies, such as augmented reality.

These are some of the points that have not been addressed in the course of this work, but which should be part of the migration guidelines of each organization. This trend indicates that Industry 4.0 vision would lead to a significant reduction in standardized low-skill and a growth in high-skill activities [57] on higher levels of the ISA-95 (*i.e.*, planning, control). Due to this, social impact is expected an increasing complexity in many job profiles. Another open issue is the IT-related jobs, what are the new required skills/roles to deal with the challenge to integrate the CPS in order to move towards a smart production? If we think about it, over the years, repetitive and intensive physical effort jobs tend to be replaced by robots and the human cognitive ability by powerful machine learning models. However, whether the human being is going to be entirely replaced by intelligent and autonomous (*i.e.*, self- * features) systems, is still an open question and its implementation is getting closer due to the industry 4.0 vision.

Ideally, the ADvISER should be tested in other domains to test its ability to change, and also to demonstrate the benefits of this architecture, which has become even more evident, the more dynamic and evolvable the system is. Testing ADvISER by scaling-up the case study to any other benchmark, (*i.e.*, more machines and services) is also being considered, to consolidate the current results further.

In the context of ADvISER, one of the main objectives is a technological architecture overlying ML algorithms. In this situation, releasing the components of the ADvISER architecture as a service to other researchers will allow not only testing the limits of their adaptation algorithms but also test the interoperability of the approach.

This work achievements regarding reconfigurability allow the creation of value over time. Regarding the previous research questions, almost all lines of research can be investigated or expanded to other areas where there is the possibility of adaptation, and even in some cases, beyond the manufacturing industry.

APPENDIX A

APPENDIX A

This Appendix lists the most important and significant scientific contributions published during the work on this thesis, and all the contributions are from international, IEEE sponsored peer review conferences indexed either to the Scopus or Web of Science.

List of Journal Publications:

The articles listed below were published in **Q1** journals according to the Scimago Journal Ranking.

List of Journal Publications:

- N. Rodrigues, E. Oliveira, P. Leitão, **Decentralized and on-the-fly agent-based service reconfiguration in manufacturing systems**, Computers in Industry, 101 (2018), pp. 81–90.
- P. Leitão, N. Rodrigues, C. Turrin, A. Pagani, **Multiagent system integrating process and quality control in a factory producing laundry washing machines**, IEEE Transactions on Industrial Informatics, 11 (2015), pp. 879–886.
- P. Leitão, N. Rodrigues, J. Barbosa, C. Turrin, A. Pagani, **Intelligent products: The grace experience**, Control Engineering Practice, 42 (2015), pp. 95–105.

List of Book Chapters:

- P. Petrali, A. Pagani, F. Boschi, G. Tavola, P. Fantini, J. Barbosa, N. Rodriques, A. Ferreira, G. Angione "Use Case: White Goods" In Colombo et al. (Eds.). In Digitalized and Harmonized Industrial Production Systems: The PERFoRM Approach 2019.
- P. Leitão, N. Rodrigues, C. Turrin, A. Pagani, Multi-agent System for Integrating Quality and Process Control in a Home Appliance Production Line In Leitão, P., & Karnouskos, S. (Eds.). Industrial Agents: Emerging Applications of Software Agents in Industry. Morgan Kaufmann 2015.

List of Conference Publications (Indexed Scopus or Web of Science):

1. N. Rodrigues, P. Leitão, E. Oliveira, An agent-based approach for the dynamic and decentralized service reconfiguration in collaborative production scenarios, in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 10444 LNAI, Springer, Cham, 2017, pp. 140–154.
2. N. Rodrigues, P. Leitão, E. Oliveira, Dynamic Service Reconfiguration with multi-agent Systems, in Service Orientation in Holonic and Multi-Agent Manufacturing, Springer, Cham, 2017, pp. 307–318
3. N. Rodrigues, P. Leitão, E. Oliveira, Triggering strategies for automatic and online service reconfiguration, in Iberian Conference on Information Systems and Technologies, CISTI, vol. 2016-July, Gran Canaria, Spain, 2016.
4. N. Rodrigues, P. Leitão, and E. Oliveira, Adaptive Services Reconfiguration in Manufacturing Environments Using a Multi-agent System Approach, in Multiagent, In MATES 2015: Smart Things Working Together, Jörg P.Müller, Wolf Ketter, Gal Kaminka, Gerd Wagner, and Nils Bulling, Cottbus, Germany, 2015.
5. P. Leitão, N. Rodrigues, and J. Barbosa, What-if game simulation in agentbased strategic production planners, in IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, vol. 2015-Octob, 2015.
6. Ferreira, A. Pereira, N. Rodrigues, J. Barbosa, P. Leitão, Integration of an agent-based strategic planner in an enterprise service bus ecosystem, in Proceeding - 2015 IEEE International Conference on Industrial Informatics, INDIN2015, 2015.
7. N. Rodrigues, P. Leitão, and E. Oliveira, Self-Interested Service-Oriented Agents based on Trust and QoS for Dynamic Reconfiguration, in Service Orientation in Holonic and Multi-Agent Manufacturing (SOHOMA'14), Nice, France, 2014.
8. N. Rodrigues, P. Leitão, and E. Oliveira, Dynamic Composition of Service Oriented Multi-agent System in Self-organized Environments, in IAT4SIS '14, Prague, Czech Republic, 2014.
9. N. Rodrigues, E. Oliveira, and P. Leitão, Self-organization Combining Incentives and Risk Management for a Dynamic Service-Oriented Multiagent System, in Technological Innovation for Collective Awareness.
10. D. Costa, F. Pires, N. Rodrigues, J. Barbosa, G. Igreijas, P. Leitão, Empowering Humans in a Cyber-Physical Production System: Human-in-the-loop Prespective, IEEE International Conference on Industrial Cyber-Physical Systems (ICPS 2019), IEEE, 5 2019, pp. 139–144.
11. Pereira, N. Rodrigues, J. Barbosa, P. Leitão, Trust and Risk Management Towards Resilient Large-scale Cyber-Physical Systems, in Proceedings of the IEEE International Symposium on Industrial Electronics (ISIE 2013), 2013.
12. Pereira, N. Rodrigues, and P. Leitão, Data collection for global monitoring and trend analysis in the GRACE multi-agent system, in Industrial Technology (ICIT), 2013 IEEE International Conference on, pp. 1240–1245, 2013.

13. N. Rodrigues, P. Leitão, M. Foehr, C. Turrin, A. Pagani, and R. Decesari, Adaptation of functional inspection test plan in a production line using a multiagent system, in Industrial Electronics (ISIE), 2013 IEEE International Symposium on, pp. 1–6, 2013.
14. N. Rodrigues, A. Pereira, and P. Leitão, Adaptive Multi-Agent System for a Washing Machine Production Line, in Industrial Applications of Holonic and Multi-Agent Systems SE - 19, vol. 8062, V. Maríık, J. M. Lastra, and P. Skobelev, Eds. Springer Berlin Heidelberg, pp. 212–223, 2013.
15. P. Leitão, N. Rodrigues, Modelling and Validating the Multi-agent System Behaviour for a Washing Machine Production Line, Proceedings of the IEEE International Symposium on Industrial Electronics (ISIE´12), Hangzhou, China, 28-31 May 2012. doi: 10.1109/ISIE.2012.6237260.
16. Pereira, N. Rodrigues, P. Leitão, Deployment of Multi-agent Systems for Industrial Application, Emerging Technologies on Factory Automation (ETFA'12) Kraków, Poland, September 2012.
17. L. Stroppa, N. Rodrigues, P. Leitão, N. Paone, Quality Control Agents for Adaptive Visual Inspection in Production Lines, IEEE Industrial Electronics Society (IECON'12) Montréal, Canada, October 2012.
18. P. Leitão, N. Rodrigues, C. Turrin , A. Pagani ,P. Petrali, GRACE Ontology Integrating Process and Quality Control, IEEE Industrial Electronics Society (IECON'12) Montréal, Canada, October 2012.
19. P. Leitão, N. Rodrigues, Multi-agent system for on-demand production integrating production and quality control, Holonic and Multi-Agent Systems for Manufacturing (HOLOMAS'11) (V. Marik, P. Vrba, and P. Leitao, eds.), Lecture Notes in Computer Science, vol. 6867, pp. 84-93, Springer Berlin / Heidelberg, 2011. doi: 10.1007/978-3-642-23181-0_8.

[1] *2660.1 - Recommended Practices on Industrial Agents: Integration of Software Agents and Low Level Automation Functions*.

[2] *Autonomous Systems and Artificial Life*, in Ubiquitous Computing, John Wiley & Sons, Ltd, Chichester, UK, pp. 317–341.

[3] *IBM's Perspective on the State of Information Technology*, tech. rep.

[4] *VDI/VDE 2653 Sheet 1, 2 and 3: multi-agent systems in industrial automation: Fundamentals, Development and Application*, 2010.

[5] *Specification of the multi-agent architecture for line-production system, integrating process and quality control*, tech. rep., IPB, UNIVPM, SINTEF, Whirlpool, AEA, SIEMENS, 2011.

[6] *D2.4 Analysis of the State-of-the-Art and Future Challenges in Cyber-physical Systems of Systems*, (2013).

[7] Y. ALSAFI AND V. VYATKIN, *Ontology-based reconfiguration agent for intelligent mechatronic systems in flexible manufacturing*, Robotics and Computer-Integrated Manufacturing, 26 (2010), pp. 381–391.

[8] J. ANTÓNIO, *Coalition Based Approach for Shop Floor Agility-A Multiagent Approach*, PhD thesis, Universidade Nova de Lisboa, 2003.

[9] W. R. ASHBY, *Principles of the self-organizing dynamic system*, The Journal of general psychology, 37 (1947), pp. 125–128.

[10] C. BĂDICĂ, Z. BUDIMAC, H.-D. BURKHARD, AND M. IVANOVIC, *Software agents: Languages, tools, platforms*, Computer Science and Information Systems, 8 (2011), pp. 255–298.

[11] Y. BAR-YAM, *Dynamics of complex systems*, 2003.

[12] C. J. BARTODZIEJ, *The concept Industry 4.0*, in The Concept Industry 4.0, Springer Fachmedien Wiesbaden, Wiesbaden, 2017, pp. 27–50.

[13] H. BAUER, C. BAUR, GIANLUCA CAMPLONE, AND ET. AL., *Industry 4.0: How to navigate digitization of the manufacturing sector*, tech. rep., McKinsey Digital, 2015.

[14] L. S. BELISARIO AND H. PIERREVAL, *A conceptual framework for analyzing adaptable and reconfigurable manufacturing systems*, in Industrial Engineering and Systems Management (IESM), Proceedings of 2013 International Conference on, 2013, pp. 1–7.

[15] F. L. BELLIFEMINE, G. CAIRE, AND D. GREENWOOD, *Developing Multi-Agent Systems with JADE (Wiley Series in Agent Technology)*, John Wiley & Sons, 2007.

[16] D. BHATIA, *Reconfigurable computing*, Proceedings Tenth International Conference on VLSI Design, (1997), pp. 356–359.

[17] Z. M. BI, S. Y. T. LANG, W. SHEN, AND L. WANG, *Reconfigurable manufacturing systems: the state of the art*, International Journal of Production Research, 46 (2008), pp. 967–992.

[18] T. BORANGIU, S. RĂILEANU, D. TRENTESAUX, AND T. BERGER, *Open manufacturing control with agile reconfiguring of resource services*, Control Engineering and Applied Informatics, 12 (2010), pp. 10–17.

[19] M. BRUCCOLERI *, P. RENNA, AND G. PERRONE, *Reconfiguration: a key to handle exceptions and performance deteriorations in manufacturing operations*, International Journal of Production Research, 43 (2005), pp. 4125–4145.

[20] G. C. BUTTAZZO, *Hard RealTime Computing Systems*, Springer New York Dordrecht Heidelberg London, 2011.

[21] A. CALA, A. LUDER, F. BOSCHI, G. TAVOLA, AND M. TAISCH, *Migration towards digital manufacturing automation — An assessment approach*, in 2018 IEEE Industrial Cyber-Physical Systems (ICPS), IEEE, 5 2018, pp. 714–719.

[22] G. CANDIDO, C. SOUSA, G. DI ORIO, J. BARATA, AND A. W. COLOMBO, *Enhancing device exchange agility in Service-oriented industrial automation*, in 2013 IEEE International Symposium on Industrial Electronics, IEEE, 5 2013, pp. 1–6.

[23] C. CARDEIRA AND Z. MAMMERI, *A schedulability analysis of tasks and network traffic in distributed real-time systems*, Measurement, 15 (1995), pp. 71–83.

[24] O. CARDIN, W. DERIGENT, AND D. TRENTESAUX, *Evolution of holonic control architectures towards Industry 4.0: A short overview*, IFAC-PapersOnLine, 51 (2018), pp. 1243–1248.

[25] I. CASTELO-BRANCO, F. CRUZ-JESUS, AND T. OLIVEIRA, *Assessing Industry 4.0 readiness in manufacturing: Evidence for the European Union*, Computers in Industry, (2019).

[26] R. CENTENO, *Mecanismos Incentivos para la Regulación de Sistemas MultiAgente Abiertos basados en Organizacione*, PhD thesis, Universidad Rey Juan Carlos, 2012.

[27] G. B. CHAFLE, S. CHANDRA, V. MANN, AND M. G. NANDA, *Decentralized Orchestration of Composite Web Services*, in Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers &Amp; Posters, WWW Alt. '04, New York, NY, USA, 2004, ACM, pp. 134–143.

[28] V. CHANDOLA, A. BANERJEE, AND V. KUMAR, *Anomaly detection: A survey*, ACM Computing Surveys (CSUR), 41 (2009), pp. 1–58.

[29] A. COLOMBO, S. KARNOUSKOS, AND T. BANGEMANN, *Industrial Cloud-Based Cyber-Physical Systems*, Industrial Cloud-Based Cyber-Physical Systems, (2014), pp. 23–47.

[30] A. W. COLOMBO, *Development and Implementation of Hierarchical Control Structures of Flexible Production Systems Using High Level Petri Nets*, Fertigungstechnik - Erlangen, Meisenbach, 1998.

[31] A. W. COLOMBO, F. JAMMES, H. SMIT, R. HARRISON, J. L. M. LASTRA, AND I. M. DELAMER, *Service-oriented architectures for collaborative automation*, in Industrial Electronics Society, 2005. 31st Annual Conference of IEEE, 2005, p. 6 pp.

[32] A. W. COLOMBO AND S. KARNOUSKOS, *Towards the Factory of the Future: A Service-oriented Cross-layer Infrastructure*, in ICT Shaping the World: A Scientific View, no. ISBN: 9780470741306, European Telecommunications Standards Institute (ETSI), John Wiley and Sons, 2009, pp. 65–81.

[33] A. W. COLOMBO, S. KARNOUSKOS, AND T. BANGEMANN, *Industrial Cloud-Based Cyber-Physical Systems*, Industrial Cloud-Based Cyber-Physical Systems, (2014), pp. 23–47.

[34] A. W. COLOMBO, S. KARNOUSKOS, AND J. M. MENDES, *Factory of the Future: A Service-oriented System of Modular, Dynamic Reconfigurable and Collaborative Systems*, in Artificial Intelligence Techniques for Networked Manufacturing Enterprises Management, L. Benyoucef and B. Grabot, eds., Springer London, London, 2010, pp. 459–481.

[35] D. COSTA, F. PIRES, N. RODRIGUES, J. BARBOSA, G. IGREJAS, AND P. LEITAO, *Empowering Humans in a Cyber-Physical Production System: Human-in-the-loop Perspective*, in 2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS), Taipei, Taiwan, 5 2019, IEEE, pp. 139–144.

[36] F. CURBERA, R. KHALAF, N. MUKHI, S. TAI, AND S. WEERAWARANA, *The next step in Web services*, Commun. ACM, 46 (2003), pp. 29–34.

[37] W. DAI, WANQI HUANG, AND V. VYATKIN, *Enabling plug-and-play software components in industrial cyber-physical systems by adopting service-oriented architecture paradigm*, in IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, IEEE, 10 2016, pp. 5253–5258.

[38] C. DARWIN, *On the Origin of the Species*, 1859.

[39] O. L. DE WECK, A. M. ROSS, AND D. H. RHODES, *Investigating Relationships and Semantic Sets amongst System Lifecycle Properties (Ilities)*, tech. rep., Delft University of Technology, Delft, 2012.

[40] E. DEL VAL, M. REBOLLO, AND V. BOTTI, *Combination of self-organization mechanisms to enhance service discovery in open systems*, Information Sciences, 279 (2014), pp. 138–162.

[41] I. M. DELAMER AND J. L. MARTINEZ LASTRA, *Self-orchestration and choreography: Towards architecture-agnostic manufacturing systems*, in Proceedings - International Conference on Advanced Information Networking and Applications, AINA, vol. 2, 2006, pp. 573–577.

[42] J. DIAS-FERREIRA, L. RIBEIRO, H. AKILLIOGLU, P. NEVES, AND M. ONORI, *BIOSOARM: a bio-inspired self-organising architecture for manufacturing cyber-physical shopfloors*, Journal of Intelligent Manufacturing, 29 (2018), pp. 1659–1682.

[43] E. W. DIJKSTRA, *Programming considered as a Human Activity*, in Proceedings of IFIP ´65, 1965.

[44] A. DIONÍSIO BETTENCOURT DA SILVA PARREIRA ROCHA, *Increase the adoption of Agent-based Cyber-Physical Production Systems through the Design of Minimally Invasive Solutions*, PhD thesis, 2018.

[45] S. DOBSON, F. ZAMBONELLI, S. DENAZIS, A. FERNÁNDEZ, D. GAÏTI, E. GELENBE, F. MASSACCI, P. NIXON, F. SAFFRE, AND N. SCHMIDT, *A survey of autonomic communications*, ACM Transactions on Autonomous and Adaptive Systems, 1 (2006), pp. 223–259.

[46] S. C. DOLTSINIS, S. RATCHEV, AND N. LOHSE, *A Framework for Performance Measurement during Production Ramp-up of Assembly Stations*, European Journal of Operational Research, 229 (2013), pp. 85–94.

[47] L. M. DOOLEY, *Case Study Research and Theory Building*, Advances in Developing Human Resources, 4 (2002), pp. 335–354.

[48] L. DUERKOP, H. TRSEK, J. JASPERNEITE, AND L. WISNIEWSKI, *Towards autoconfiguration of industrial automation systems: A case study using Profinet IO*, Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012), (2012), pp. 1–8.

[49] E. N. EIDE, *Software variability mechanisms for improving run-time performance*, ProQuest Dissertations and Theses, 3547966 (2012), p. 116.

[50] H. A. EL MARAGHY, *Flexible and reconfigurable manufacturing systems paradigms*, in Flexible Services and Manufacturing Journal, vol. 17, 2006, pp. 261–276.

[51] T. ERL, *Service-Oriented Architecture: Concepts, Technology, and Design*, 2005.

[52] B. ESMAEILIAN, S. BEHDAD, AND B. WANG, *The evolution and future of manufacturing: A review*, Journal of Manufacturing Systems, 39 (2016), pp. 79–100.

[53] C. EUROPE, *Production harmonizEd Reconfiguration of Flexible Robots and Machinery*, 2015.

[54] EUROPEAN COMMISSION, *Technology readiness levels (TRL)*, tech. rep., 2017.

[55] E. EUROPEAN COMMISSION, *Factories of the future strategic multi-annual roadmap*, tech. rep., Directorate-General for Research and Innovation (European Commission) , European Factories of the Future Research Association (EFFRA), Brussels, 2013.

[56] A. EYTAN, *openMOS Open dynamic Manufacturing Operating System for Smart Plug-and-Produce Automation Components*, tech. rep., We Plus S.r.l. (Italy), 2016.

[57] P. FANTINI, M. PINZONE, AND M. TAISCH, *Placing the operator at the centre of Industry 4.0 design: Modelling and assessing human activities within cyber-physical systems*, Computers & Industrial Engineering, (2018), p. 105058.

[58] A. M. FARID, *Measures of reconfigurability and its key characteristics in intelligent manufacturing systems*, Journal of Intelligent Manufacturing, (2014), pp. 1–17.

[59] J. FERBER, *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*, Addison-Wesley Longman Publishing Co., Boston, MA, USA, 1st ed., 1999.

[60] J. FERBER AND O. GUTKNECHT, *Aalaadin: a meta-model for the analysis and design of organizations in multi-agent systems*, in Proceedings of the Third International Conference on Multi-Agent Systems, ICMAS'98, Y. Demazeau, ed., Paris, France, 1998, IEEE Computer Society, pp. 128–135.

[61] A. FERNANDES, *Génese e dinâmica atual do Conceito "Indústria 4.0": Uma abordagem bibliométrica*, tech. rep., ISCTE Business School, Lisbon, 2018.

[62] A. FERREIRA, A. PEREIRA, N. RODRIGUES, J. BARBOSA, AND P. LEITÃO, *Integration of an agent-based strategic planner in an enterprise service bus ecosystem*, in Proceeding - 2015 IEEE International Conference on Industrial Informatics, INDIN 2015, 2015.

[63] P. FERREIRA, S. DOLTSINIS, AND N. LOHSE, *Symbiotic Assembly Systems - A New Paradigm*, Procedia CIRP, 17 (2014), pp. 26–31.

[64] T. FININ, R. FRITZSON, D. MCKAY, AND R. MCENTIRE, *KQML as an agent communication language*, in Proceedings of the third international conference on Information and knowledge management, CIKM '94, New York, USA, 1994, ACM, pp. 456–463.

[65] FIPA, *FIPA-Contract Net Interaction Protocol Specification*, 2001.

[66] M. FOEHR, T. JÄGER, C. TURRIN, P. PETRALI, AND A. PAGANI, *Methodology for consideration of product quality within factory automation engineering*, in Proceedings of the IEEE International Conference on Industrial Technology, 2013, pp. 1333–1338.

[67] M. FOEHR, J. VOLLMAR, A. CALÀ, P. LEITÃO, S. KARNOUSKOS, AND A. W. COLOMBO, *Engineering of Next Generation Cyber-Physical Automation System Architectures*, in Multi-Disciplinary Engineering for Cyber-Physical Production Systems: Data Models and Software Solutions for Handling Complex Engineering Projects, S. Biffl, A. Lüder, and D. Gerhard, eds., Springer International Publishing, Cham, 2017, pp. 185–206.

[68] FOUNDATION FOR INTELLIGENT PHYSICAL AGENTS, *The foundation for intelligent physical agents*, 2013.

[69] S. FRANKLIN AND A. GRAESSER, *Is It an Agent, or Just a Program?: A Taxonomy for Autonomous Agents*, in Proceedings of the Workshop on Intelligent Agents III, Agent Theories, Architectures, and Languages, ECAI '96, London, UK, UK, 1997, Springer-Verlag, pp. 21–35.

[70] F. GAMBOA QUINTANILLA, O. CARDIN, A. L'ANTON, AND P. CASTAGNA, *A modeling framework for manufacturing services in Service-oriented Holonic Manufacturing Systems*, Engineering Applications of Artificial Intelligence, 55 (2016), pp. 26–36.

[71] H. M. GASPAR, A. M. ROSS, D. H. RHODES, AND S. O. ERIKSTAD, *Handling Complexity Aspects in Conceptual Ship Design*, Proceedings of the 11th International Marine Design Conference, (2012).

[72] A. GIRET, E. GARCIA, AND V. BOTTI, *An engineering framework for Service-Oriented Intelligent Manufacturing Systems*, Computers in Industry, 81 (2016), pp. 116–127.

[73] G. GONÇALVES, *Enhancing life cycle sustainability in system of systems: an event driven framework for changeability*, PhD thesis, University of Porto, 11 2015.

[74] G. GONÇALVES, J. REIS, R. PINTO, M. ALVES, AND J. CORREIA, *A Step Forward on Intelligent Factories: A Smart Sensor-oriented Approach*, Proceedings of the IEEE International Conference on Emerging Technology and Factory Automation (ETFA'14), (2014), p. 1–8.

[75] D. N. GUJARATI AND D. C. PORTER, *Basic Econometrics*, Economics series, McGraw-Hill Irwin, 2009.

[76] M. HARMAN, E. BURKE, J. CLARK, AND X. YAO, *Dynamic adaptive search based software engineering*, in Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement - ESEM '12, New York, New York, USA, 2012, ACM Press, p. 1.

[77] M. HERMANN, T. PENTEK, AND B. OTTO, *Design Principles for Industrie 4.0 Scenarios: A Literature Review*, Technische Universitat Dortmund, (2015).

[78] J. HIELSCHER, A. METZGER, AND R. KAZHAMIAKIN, *Techniques and Methodologies for Monitoring and Adaptation of SBAs*, tech. rep., 2009.

[79] M. C. HUEBSCHER AND J. A. MCCANN, *A survey of Autonomic Computing-degrees, models and applications*, tech. rep.

[80] B. L. HUTCHINGS AND M. J. WIRTHLIN, *Implementation Approaches for Reconfigurable Logic Applications*, in Proc. Int. Conf. on Field Programmable Logic and Applications (FPL), vol. 975, 1995, pp. 419–428.

[81] F. JAENSCH, A. CSISZAR, C. SCHEIFELE, AND A. VERL, *Digital Twins of Manufacturing Systems as a Base for Machine Learning*, in Proceedings of the 2018 25th International Conference on Mechatronics and Machine Vision in Practice, M2VIP 2018, Institute of Electrical and Electronics Engineers Inc., 1 2019.

[82] T. JUAN, A. PEARCE, AND L. STERLING, *ROADMAP: Extending the Gaia Methodology for Complex Open Systems*, ACM Press, 2002, pp. 3–10.

[83] H. KAGERMANN, W. WOLFGANG, AND J. HELBIG, *Securing the future of German manufacturing industry. Recommendations for implementing the strategic initiative INDUSTRIE 4.0. Final report of the Industrie 4.0 Working Group*, tech. rep., 2013.

[84] S. KARNOUSKOS, A. W. COLOMBO, T. BANGEMANN, K. MANNINEN, R. CAMP, M. TILLY, M. SIKORA, F. JAMMES, J. DELSING, J. ELIASSON, P. NAPPEY, J. HU, AND M. GRAF, *The IMC-AESOP Architecture for Cloud-Based Industrial Cyber-Physical Systems*, in Industrial Cloud-Based Cyber-Physical Systems, Springer International Publishing, Cham, 2014, pp. 49–88.

[85] J. O. KEPHART AND D. M. CHESS, *The vision of autonomic computing*, Computer, 36 (2003), pp. 41–50.

[86] A. KHELIL, A. BROE-RING, D. ANICIC, D. HERRLING, J. ZIBUSCHKA, J. LIM, K. REHFELDT, S. SCHMID, S. KAEBISCH, T. SCHÖFTNER, V. CHARPENAY, Y. WANG, AND M. SERRANO, *BIG-IOT Analysis of Technology Readiness* , tech. rep., 2016.

[87] M. KLOPPMANN, D. KOENIG, F. LEYMANN, A. RICKAYZEN, C. VON RIEGEN, P. SCHMIDT, AND I. TRICKOVIC, *WS-BPEL Extension for People - BPEL4People*, tech. rep., IBM / SAP, 2005.

[88] V. KOCH, S. KUGE, R. GEISSBAUER, AND S. SCHRAUF, *Industry 4.0 - Opportunities and challenges of the industrial internet*, strategy& Formerly Booz & Company, PwC, (2014).

[89] Y. KOREN, U. HEISEL, F. JOVANE, T. MORIWAKI, G. PRITSCHOW, G. ULSOY, AND H. VAN BRUSSEL, *Reconfigurable Manufacturing Systems*, CIRP Annals - Manufacturing Technology, 48 (1999), pp. 527–540.

[90] Y. KOREN AND M. SHPITALNI, *Design of reconfigurable manufacturing systems*, Journal of Manufacturing Systems, 29 (2010), pp. 130–141.

[91] R. KOTA, N. GIBBINS, AND N. R. JENNINGS, *Self-Organising Agent Organisations*, in The 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS '09), 2009, pp. 797–804.

[92] J. KRAMER AND J. MAGEE, *Dynamic Configuration for Distributed Systems*, IEEE Transactions on Software Engineering, SE-11 (1985), pp. 424–436.

[93] J. KREPS, *Questioning the Lambda Architecture – O Reilly*, 2014.

[94] P. LALANDA, J. A. MCCANN, AND A. DIACONESCU, *Autonomic Computing Principles, Design and Implementation*, no. April, 2013.

[95] R. LANDERS, B.-K. MIN, AND Y. KOREN, *Reconfigurable Machine Tools*, CIRP Annals - Manufacturing Technology, 50 (2001), pp. 269–274.

[96] J. L. M. LASTRA AND M. DELAMER, *Semantic web services in factory automation: Fundamental insights and research roadmap*, IEEE Transactions on Industrial Informatics, 2 (2006), pp. 1–11.

[97] F. LÉCUÉ, Y. GORRONOGOITIA, R. GONZALEZ, M. RADZIMSKI, AND M. VILLA, *SOA4All: An Innovative Integrated Approach to Services Composition*, in Web Services (ICWS), 2010 IEEE International Conference on, 2010, pp. 58–67.

[98] J. LEE, B. BAGHERI, AND H.-A. KAO, *A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems*, (2015).

[99] J. LEE, E. LAPIRA, B. BAGHERI, AND H.-A. KAO, *Recent advances and trends in predictive manufacturing systems in big data environment*, Manufacturing Letters, 1 (2013), pp. 38–41.

[100] P. LEITÃO, J. BARBOSA, A. PEREIRA, J. BARATA, AND A. W. COLOMBO, *Specification of the PERFoRM architecture for the seamless production system reconfiguration*, in IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, IEEE, 10 2016, pp. 5729–5734.

[101] P. LEITAO, J. BARBOSA, A. PEREIRA, J. BARATA, AND A. W. COLOMBO, *Specication of the PERFoRM Architecture for the Seamless Production System Reconguration*, in 2016 42nd Annual Conference of the IEEE Industrial Electronics Society (IECON), 2016.

[102] P. LEITÃO, A. W. COLOMBO, AND S. KARNOUSKOS, *Industrial automation based on cyber-physical systems technologies: Prototype implementations and challenges*, Computers in Industry, 81 (2016), pp. 11–25.

[103] P. LEITÃO AND S. KARNOUSKOS, *A Survey on Factors that Impact Industrial Agent Acceptance*, in Industrial Agents, Elsevier, 2015, pp. 401–429.

[104] P. LEITÃO, S. KARNOUSKOS, L. RIBEIRO, J. LEE, T. STRASSER, AND A. W. COLOMBO, *Smart Agents in Industrial Cyber-Physical Systems*, Proceedings of the IEEE, 104 (2016), pp. 1086–1101.

[105] P. LEITAO, V. MARIK, AND P. VRBA, *Past, Present, and Future of Industrial Agent Applications*, Industrial Informatics, IEEE Transactions on, 9 (2013), pp. 2360–2372.

[106] P. LEITÃO AND F. RESTIVO, *ADACOR: a holonic architecture for agile and adaptive manufacturing control*, Computers in Industry, 57 (2006), pp. 121–130.

[107] P. LEITÃO AND N. RODRIGUES, *Multi-agent system for on-demand production integrating production and quality control*, in Proceedings of the 5th international conference on Industrial applications of holonic and multi-agent systems for manufacturing, HoloMAS'11, Berlin, Heidelberg, 2011, Springer-Verlag, pp. 84–93.

[108] P. LEITÃO AND N. RODRIGUES, *Modelling and validating the multi-agent system behaviour for a washing machine production line*, in IEEE International Symposium on Industrial Electronics, 2012.

[109] P. LEITÃO, N. RODRIGUES, AND J. BARBOSA, *What-if game simulation in agent-based strategic production planners*, in IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, vol. 2015-Octob, 2015.

[110] P. LEITÃO, N. RODRIGUES, J. BARBOSA, C. TURRIN, AND A. PAGANI, *Intelligent products: The grace experience*, Control Engineering Practice, 42 (2015), pp. 95–105.

[111] P. LEITÃO, N. RODRIGUES, A. FERREIRA, A. PAGANI, P. PETRALI, AND J. BARBOSA, *A Lightweight Dynamic Monitoring of Operational Indicators for a Rapid Strategical Awareness*, in 2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS), IEEE, 5 2019, pp. 121–126.

[112] P. LEITÃO, N. RODRIGUES, C. TURRIN, AND A. PAGANI, *Multi-Agent System for Integrating Quality and Process Control in a Home Appliance Production Line*, 2015.

[113] P. LEITÃO, N. RODRIGUES, C. TURRIN, AND A. PAGANI, *Multi-agent System Integrating Process and Quality Control in a Factory Producing Laundry Washing Machines*, IEEE Transactions on Industrial Informatics, 11 (2015), pp. 879–886.

[114] P. LEITÃO, N. RODRIGUES, C. TURRIN, AND A. PAGANI, *Multiagent system integrating process and quality control in a factory producing laundry washing machines*, IEEE Transactions on Industrial Informatics, 11 (2015), pp. 879–886.

[115] P. LEITAO, N. RODRIGUES, C. TURRIN, A. PAGANI, AND P. PETRALI, *GRACE ontology inteGrating pRocess and quAlity Control*, in IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society, 2012, pp. 4348–4353.

[116] F. LEYMANN, *WSFL. Web services flow language*, tech. rep., 2001.

[117] L. LIAO AND J. LEE, *Design of a reconfigurable prognostics platform for machine tools*, Expert Systems with Applications, (2010).

[118] Q. LIU, M. DONG, F. F. CHEN, W. LV, AND C. YE, *Single-machine-based joint optimization of predictive maintenance planning and production scheduling*, Robotics and Computer-Integrated Manufacturing, (2019).

[119] S. LJASENKO, P. FERREIRA, L. JUSTHAM, AND N. LOHSE, *Decentralised vs partially centralised self-organisation model for mobile robots in large structure assembly*, Computers in Industry, 104 (2019), pp. 141–154.

[120] A. LÜDER, J. PESCHKE, T. SAUTER, S. DETER, AND D. DIEP, *Distributed intelligence for plant automation based on multi-agent systems: The PABADIS approach*, Production Planning and Control, 15 (2004), pp. 201–212.

[121] C. MACKENZIE, K. LASKEY, F. MCCABE, P. BROWN, AND R. METZ, *Reference model for service oriented architecture 1.0*, OASIS Standard, 12 (2006), pp. 1–28.

[122] A. MADNI, C. MADNI, AND S. LUCERO, *Leveraging Digital Twin Technology in Model-Based Systems Engineering*, Systems, 7 (2019), p. 7.

[123] D. MARCOS-JORQUERA, F. MACIA-PEREZ, V. GILART-IGLESIAS, AND J. A. GIL-MARTINEZ-ABARCA, *Service model for the management of industrial environments. Dynamic reconfiguration of production elements*, in 2007 5th IEEE International Conference on Industrial Informatics, 2007, pp. 249–254.

[124] V. MAŘÍK AND D. MCFARLANE, *Industrial adoption of agent-based technologies*, 2005.

[125] C. A. MARÍN, L. MÖNCH, P. LEITÃO, P. VRBA, D. KAZANSKAIA, V. CHEPEGIN, L. LIU, AND N. MEHANDJIEV, *A Conceptual Architecture Based on Intelligent Services for Manufacturing Support Systems*, Proceedings of the 2013 IEEE International Conference on Systems, Man, and Cybernetics, SMC, (2013), pp. 4749–4754.

[126] E. M. MAXIMILIEN AND M. P. SINGH, *A Framework and Ontology for Dynamic Web Services Selection*, IEEE Internet Computing, 8 (2004), pp. 84–93.

[127] G. MAY, B. STAHL, AND M. TAISCH, *Energy Management in Manufacturing: Toward Eco-factories of the Future – A Focus Group Study*, Applied Energy, 164 (2016), pp. 628–638.

[128] D. MCFARLANE, V. GIANNIKAS, A. C. WONG, AND M. HARRISON, *Product intelligence in industrial control: Theory and practice*, Annual Reviews in Control, 37 (2013), pp. 69–88.

[129] D. C. MCFARLANE AND S. BUSSMANN, *Developments in Holonic Production Planning and Control*, publ. in Int. Journal of Production Planning and Control, 11 (2000), pp. 522–536.

[130] D. L. MCGUINNESS AND F. VAN HARMELEN, *OWL Web Ontology Language Overview*, W3C recommendation, 10 (2004), pp. 1–22.

[131] J. MENDES, *Engineering Framework for Service-oriented Automation Systems*, PhD thesis, Faculdade de Engenharia da Universidade do Porto, 2011.

[132] G. G. MEYER AND J. HOLMSTRÖM, *Intelligent Products: A survey*, Computers in Industry, 60 (2009), pp. 137–148.

[133] I. MEZGAR AND U. RAUSCHECKER, *The Challenge of Networked Enterprises for Cloud Computing Interoperability*, Computers in Industry, 65 (2014), p. 657–674.

[134] T. M. MITCHELL, *Machine Learning*, McGraw-Hill, New York, 1997.

[135] F. MORAES, N. CALAZANS, L. MÖLLER, E. BRIÃO, AND E. CARVALHO, *Dynamic and partial reconfiguration in FPGA SoCs: Requirements tools and a case study*, in New Algorithms, Architectures and Applications for Reconfigurable Computing, 2005, pp. 157–168.

[136] J. P. MÜLLER AND K. FISCHER, *Application Impact of Multi-agent Systems and Technologies: A Survey*, in Agent-Oriented Software Engineering, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014, pp. 27–53.

[137] L. MURPHY AND P. EDWARDS, *Bridging the Valley of Death: Transitioning from Public to Private Sector Financing*, tech. rep., National Renewable Energy Laboratory Golden, 2003.

[138] M. R. NAMI AND K. BERTELS, *A Survey of Autonomic Computing Systems*, in Third International Conference on Autonomic and Autonomous Systems (ICAS'07), IEEE, 6 2007, pp. 26–26.

[139] NATIONAL SCIENCE FOUNDATION (NSF), *Cyber-Physical Systems (CPS)*, 2014.

[140] P. NEVES, *Reconfiguration Methodology to improve the agility and sustainability of Plug and Produce Systems*, PhD thesis, 2016.

[141] L. OBRST, *Ontologies for semantically interoperable systems*, in Proceedings of the twelfth international conference on Information and knowledge management, CIKM '03, New York, USA, 2003, ACM, pp. 366–369.

[142] E. OLIVEIRA, K. FISCHER, AND O. STEPANKOVA, *Multi-agent systems: Which research for which applications*, Robotics and Autonomous Systems, 27 (1999), pp. 91–106.

[143] N. OLIVEIRA, *Architectural reconfiguration of interacting services*, PhD thesis, University of Minho, 1 2015.

[144] M. ONORI, N. LOHSE, J. BARATA, AND C. HANISCH, *The IDEAS project: plug & produce at shop-floor level*, Assembly Automation, 32 (2012), pp. 124–134.

[145] F. OSAREH, *Bibliometrics, citation analysis and co-citation analysis: A review of literature I*, Libri, (1996).

[146] C. PACH, T. BERGER, T. BONTE, AND D. TRENTESAUX, *ORCA-FMS: A dynamic architecture for the optimized and reactive control of flexible manufacturing scheduling*, Computers in Industry, 65 (2014), pp. 706–720.

[147] M. P. PAPAZOGLOU, P. TRAVERSO, S. DUSTDAR, AND F. LEYMANN, *Service-Oriented Computing: State of the Art and Research Challenges*, IEEE Computer Society, (2007).

[148] A. K. PARLIKAD AND D. MCFARLANE, *RFID-based product information in end-of-life decision making*, Control Engineering Practice, 15 (2007), pp. 1348–1363.

[149] M. PĚCHOUČEK AND V. MAŘÍK, *Industrial deployment of multi-agent technologies: review and selected case studies*, Autonomous Agents and Multi-Agent Systems, 17 (2008), pp. 397–431.

[150] A. PEREIRA, N. RODRIGUES, J. BARBOSA, AND P. LEITÃO, *Trust and Risk Management Towards Resilient Large-scale Cyber-Physical Systems*, in Proceedings of the IEEE International Symposium on Industrial Electronics (ISIE 2013), Taipei, 2013.

[151] A. PEREIRA, N. RODRIGUES, AND P. LEITÃO, *Deployment of Multi-agent Systems for Industrial Application*, Emerging Technologies on Factory Automation(ETFA2012), (2012).

[152] A. PEREIRA, N. RODRIGUES, AND P. LEITÃO, *Data collection for global monitoring and trend analysis in the GRACE multi-agent system*, in Industrial Technology (ICIT), 2013 IEEE International Conference on, IEEE, 2013, pp. 1240–1245.

[153] R. S. PERES, A. DIONISIO ROCHA, P. LEITAO, AND J. BARATA, *IDARTS – Towards intelligent data analysis and real-time supervision for industry 4.0*, Computers in Industry, (2018).

[154] P. PETRALI, A. PAGANI, F. BOSCHI, G. TAVOLA, P. FANTINI, J. BARBOSA, N. RODRIGUES, A. FERREIRA, AND G. ANGIONE, *Use Case: White Goods*, Digitalized and Harmonized Industrial Production Systems: The PERFoRM Approach, (2019), p. 213.

[155] R. W. PICARD, *Affective Computing MIT Technical Report #321.*, tech. rep., 1995.

[156] J. PUTTONEN, A. LOBOV, M. A. C. SOTO, AND J. L. M. LASTRA, *Cloud computing as a facilitator for web service composition in factory automation*, Journal of Intelligent Manufacturing, (2016).

[157] L. QIN AND Q. LI, *A New Construction of Job-Shop Scheduling System Integrating ILOG and MAS*, Journal of Software, 7 (2012).

[158] R CORE TEAM, *R: A Language and Environment for Statistical Computing*, 2015.

[159] G. REINHART, S. KRUG, S. HUTTNER, Z. MARI, F. RIEDELBAUCH, AND M. SCHLÖGEL, *Automatic configuration (Plug & Produce) of Industrial Ethernet networks*, in 9th IEEE/IAS International Conference on Industry Applications, 2010.

[160] L. RIBEIRO AND M. HOCHWALLNER, *On the design complexity of cyberphysical production systems*, Complexity, 2018 (2018).

[161] L. RIBEIRO, A. ROCHA, A. VEIGA, AND J. BARATA, *Collaborative routing of products using a self-organizing mechatronic agent framework—A simulation study*, Computers in Industry, 68 (2015), pp. 27–39.

[162] ROAD2CPS, *Road2CPS Technology and Application Roadmap (D2.3)*, tech. rep., 2015.

[163] A. ROCHA, G. D. ORIO, J. BARATA, N. ANTZOULATOS, E. CASTRO, D. SCRIMIERI, S. RATCHEV, AND L. RIBEIRO, *An agent based framework to support plug and produce*, in IEEE International Conference on Industrial Informatics, 2014, pp. 504–510.

[164] N. RODRIGUES, *Development of an ontology for a multi-agent system controlling a production line*, PhD thesis, Instituto Politécnico de Bragança, Escola Superior, Bragança, 2012.

[165] N. RODRIGUES, P. LEITÃO, M. FOEHR, C. TURRIN, A. PAGANI, AND R. DECESARI, *Adaptation of functional inspection test plan in a production line using a multi-agent system*, in Industrial Electronics (ISIE), 2013 IEEE International Symposium on, IEEE, 2013, pp. 1–6.

[166] N. RODRIGUES, P. LEITÃO, AND E. OLIVEIRA, *Dynamic Composition of Service Oriented Multi-agent System in Self-organized Environments*, in Proceedings of the 2014 Workshop on Intelligent Agents and Technologies for Socially Interconnected Systems - IAT4SIS '14, vol. 18August, Prague, Czech Republic, 2014, ACM Press, pp. 1–6.

[167] N. RODRIGUES, P. LEITAO, AND E. OLIVEIRA, *Adaptive Services Reconfiguration in Manufacturing Environments Using a Multi-agent System Approach*, in Multiagent System Technologies : 13th German Conference, MATES 2015, Cottbus, Germany, September 28 - 30, 2015, Revised Selected Papers, P. J. Müller, W. Ketter, G. Kaminka, G. Wagner, and N. Bulling, eds., Springer International Publishing, Cham, 2015, pp. 280–284.

[168] N. RODRIGUES, P. LEITÃO, AND E. OLIVEIRA, *Self-interested service-oriented agents based on trust and QoS for dynamic reconfiguration*, vol. 594, Nice, France, 2015.

[169] N. RODRIGUES, P. LEITÃO, AND E. OLIVEIRA, *Service reconfiguration in dynamic environments: a service-oriented multi-agent system approach*, Ciencia 2016, (2016).

[170] N. RODRIGUES, P. LEITAO, AND E. OLIVEIRA, *Triggering strategies for automatic and online service reconfiguration*, in Iberian Conference on Information Systems and Technologies, CISTI, vol. 2016-July, Gran Canaria, Spain, 2016.

[171] N. RODRIGUES, P. LEITÃO, AND E. OLIVEIRA, *An agent-based approach for the dynamic and decentralized service reconfiguration in collaborative production scenarios*, in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 10444 LNAI, Springer, Cham, 2017, pp. 140–154.

[172] N. RODRIGUES, P. LEITAO, AND E. OLIVEIRA, *Dynamic Service Reconfiguration with Multi-agent Systems*, in Service Orientation in Holonic and Multi-Agent Manufacturing, Springer, Cham, 2017, pp. 307–318.

[173] N. RODRIGUES, E. OLIVEIRA, AND P. LEITÃO, *Self-organization Combining Incentives and Risk Management for a Dynamic Service-Oriented Multi-agent System*, in Technological Innovation for Collective Awareness Systems, Springer Berlin Heidelberg, Lisboa, Portugal, 2014, pp. 101–108.

[174] N. RODRIGUES, E. OLIVEIRA, AND P. LEITÃO, *Managing dynamic reconfigurations in smart manufactures using multi-agent systems*, in V Encontro de Jovens Investigadores do Instituto Politécnico de Bragança, Instituto Politécnico de Bragança, Bragança, 2017.

[175] N. RODRIGUES, E. OLIVEIRA, AND P. LEITAO, *Decentralized and on-the-fly agent-based service reconfiguration in manufacturing systems*, Computers in Industry, 101 (2018), pp. 81–90.

[176] N. RODRIGUES, A. PEREIRA, AND P. LEITÃO, *Adaptive Multi-Agent System for a Washing Machine Production Line*, in Industrial Applications of Holonic and Multi-Agent Systems SE - 19, V. Mařík, J. Lastra, and P. Skobelev, eds., vol. 8062 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2013, pp. 212–223.

[177] D. ROMERO, P. BERNUS, O. NORAN, J. STAHRE, AND A. F. BERGLUND, *The operator 4.0: Human cyber-physical systems & adaptive automation towards human-automation symbiosis work systems*, in IFIP Advances in Information and Communication Technology, vol. 488, Springer, Cham, 9 2016, pp. 677–686.

[178] A. M. ROSS, J. C. BEESEMYER, D. H. RHODES, A. M. ROSS, J. C. BEESEMYER, AND D. H. RHODES, *A Prescriptive Semantic Basis for System Lifecycle Properties*, tech. rep., 2012.

[179] A. M. ROSS, D. H. RHODES, AND D. E. HASTINGS, *Defining changeability: Reconciling flexibility, adaptability, scalability, modifiability, and robustness for maintaining system lifecycle value*, Systems Engineering, 11 (2008), pp. 246–262.

[180] S. RUSSELL AND P. NORVIG, *Artificial intelligence: a modern approach (2nd edition)*, Prentice Hall, 2004.

[181] S. M. SAAD, *The reconfiguration issues in manufacturing systems*, in Journal of Materials Processing Technology, vol. 138, 2003, pp. 277–283.

[182] M. SALEHIE AND L. TAHVILDARI, *Self-adaptive software*, ACM Transactions on Autonomous and Adaptive Systems, 4 (2009), pp. 1–42.

[183] C. SANTOS, A. MEHRSAI, A. C. BARROS, M. ARAÚJO, AND E. ARES, *Towards Industry 4.0: an overview of European strategic roadmaps*, Procedia Manufacturing, (2017).

[184] M. S. SAYED, N. LOHSE, N. SØNDBERG-JEPPESEN, AND A. L. MADSEN, *SelSus: Towards a Reference Architecture for Diagnostics and Predictive Maintenance Using Smart Manufacturing Devices*, Proceedings of the 13th International Conference on Industrial Informatics (INDIN'15), (2015), pp. 1700–1705.

[185] S. SCHEIFELE, J. FRIEDRICH, A. LECHLER, AND A. VERL, *Flexible, Self-configuring Control System for a Modular Production System*, Procedia Technology, (2014), pp. 398–405.

[186] S. SCHOLZE, D. STOKIC, O. KOTTE, J. BARATA, G. D. ORIO, AND G. CANDIDO, *Reliable Self-Learning Production Systems Based on Context Aware Services*, Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC'13), (2013), pp. 4872–4877.

[187] J. R. SEARLE, *Speech Acts: An Essay in the Philosophy of Language*, Cambridge University Press, 1970.

[188] R. M. SETCHI AND N. LAGOS, *Reconfigurability and Reconfigurable Manufacturing Systems - State-of-the-art Review*, in 2nd IEEE International Conference on Industrial Informatics, 2004, pp. 529–535.

[189] W. SHEN AND D. H. NORRIE, *Agent-Based Systems for Intelligent Manufacturing: A State-of-the-Art Survey*, Knowledge and Information Systems, 1 (1999), pp. 129–156.

[190] M. P. SINGH, *Agent Communication Languages: Rethinking the Principles*, Computer, 31 (1998), pp. 40–47.

[191] H. SMALL, *Co-citation in the scientific literature: A new measure of the relationship between two documents*, Journal of the American Society for Information Science, (1973).

[192] M. T. SOMASHEKARA, D. S. GURU, H. S. NAGENDRASWAMY, AND K. S. MANJUNATHA, *Object-Oriented Programming With C++*, PHI Learning, 2012.

[193] SRI INTERNATIONAL, *Brazil Visits SRI to Discuss Its Economic Development Roadmap | SRI International*, 2015.

[194] L. STROPPA, N. RODRIGUES, P. LEITAO, AND N. PAONE, *Quality control agents for adaptive visual inspection in production lines*, in IECON Proceedings (Industrial Electronics Conference), 2012.

[195] S. THIEDE, M. JURASCHEK, AND C. HERRMANN, *Implementing Cyber-physical Production Systems in Learning Factories*, in Procedia CIRP, vol. 54, 2016, pp. 7–12.

[196] D. TRENTESAUX, C. PACH, A. BEKRAR, Y. SALLEZ, T. BERGER, T. BONTE, P. LEITÃO, AND J. BARBOSA, *Benchmarking flexible job-shop scheduling and control systems*, Control Engineering Practice, 21 (2013), pp. 1204–1225.

[197] W. T. TSAI, W. SONG, Y. CHEN, AND R. PAUL, *Dynamic system reconfiguration via service composition for dependable computing*, in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 4322 LNCS, 2007, pp. 203–224.

[198] H. VAN BRUSSEL, J. WYNS, P. VALCKENAERS, L. BONGAERTS, AND P. PEETERS, *Reference architecture for holonic manufacturing systems: PROSA*, Computers in Industry, 37 (1998), pp. 255–274.

[199] W. VAN DER HOEK AND M. WOOLDRIDGE, *On the logic of cooperation and propositional control*, Artificial Intelligence, 164 (2005), pp. 81–119.

[200] N. J. VAN ECK AND L. WALTMAN, *Visualizing Bibliometric Networks*, in Measuring Scholarly Impact, Springer International Publishing, Cham, 2014, pp. 285–320.

[201] VDI/VDE GESELLSCHAFT MESS- UND AUTOMATISIERUNGSTECHNIK, *Referenzarchitekturmodell Industrie 4.0 (RAMI4.0)*, tech. rep., 2016.

[202] B. VOGEL-HEUSER, A. FAY, I. SCHAEFER, AND M. TICHY, *Evolution of software in automated production systems: Challenges and research directions*, Journal of Systems and Software, 110 (2015), pp. 54–84.

[203] J. VOM BROCKE, A. SIMONS, B. NIEHAVES, K. RIEMER, R. PLATTFAUT, AND A. CLEVEN, *Reconstructing the giant: On the importance of rigour in documenting the literature search process*, in 17th European Conference on Information Systems, ECIS, 2009.

[204] N. VOROS AND K. MASSELOS, *System Level Design of Reconfigurable Systems-on-Chip*, Springer Publishing Company, Incorporated, 1st ed., 2010.

[205] P. VRBA, *JAVA-Based Agent Platform Evaluation*, Holonic and Multi-Agent Systems for Manufacturing, 2744 (2003), pp. 47–58.

[206] P. VRBA, P. TICHY;, V. MARIK;K, K. H. HALL, R. J. STARON, F. P. MATURANA, AND P. KADERA, *Rockwell Automation's Holonic and Multiagent Control Systems Compendium*, Trans. Sys. Man Cyber Part C, 41 (2011), pp. 14–30.

[207] G. WANG, S. H. HUANG, AND J. P. DISMUKES, *Product-driven supply chain selection using integrated multi-criteria decision-making methodology*, International Journal of Production Economics, 91 (2004), pp. 1–15.

[208] S. WANG AND K. G. SHIN, *Constructing reconfigurable software for machine control systems*, IEEE Transactions on Robotics and Automation, (2002).

[209] C. J. C. H. WATKINS, *Learning from Delayed Rewards*, PhD thesis, King's College, Cambridge, UK, 1989.

[210] C. J. C. H. WATKINS AND P. DAYAN, *Q-learning*, Machine Learning, 8 (1992), pp. 279–292.

[211] P. WEBB AND S. ASIF, *Advanced Flexible Automation Cell*, in Innovation for Sustainable Aviation in a Global Environment: Proceedings of the Sixth European Aeronautics Days, Madrid, 30 March-1 April, 2011, 2012, p. 296.

[212] G. WEISS, ed., *Multiagent systems: a modern approach to distributed artificial intelligence*, MIT Press, Cambridge, MA, USA, 1999.

[213] WHIRLPOOL, AEA, IPB, SIEMENS, SINTEF, AND UNIVPM, *Deliverable D5.2 Report on the results obtained in the field tests of GRACE MAS platform*, tech. rep., 2013.

[214] H. P. WIENDAHL, H. A. ELMARAGHY, P. NYHUIS, M. F. ZÄH, H. H. WIENDAHL, N. DUFFIE, AND M. BRIEKE, *Changeable Manufacturing - Classification, Design and Operation*, CIRP Annals - Manufacturing Technology, 56 (2007), pp. 783–809.

[215] WILFRIED LEPUSCHITZ, *Self-Reconfigurable Manufacturing Control based on Ontology-Driven Automation Agents*, PhD thesis, Technischen Universitat Wien, 2018.

[216] M. WOOLDRIDGE, *Agent-Based Computing*, Interoperable Communication Networks, 1 (1997), pp. 71–98.

[217] M. WOOLDRIDGE, *An Introduction to MultiAgent Systems*, Wiley Publishing, 2nd ed., 2009.

[218] M. WOOLDRIDGE AND N. R. JENNINGS, *Pitfalls of Agent-Oriented Development*, in Proceedings of the second international conference on Autonomous agents - AGENTS '98, 1988, pp. 285–391.

[219] M. WOOLDRIDGE AND N. R. JENNINGS, *Intelligent agents: Theory and practice*, Knowledge engineering review, 10 (1995), pp. 115–152.

[220] M. WOOLDRIDGE, N. R. JENNINGS, AND D. KINNY, *The Gaia Methodology for Agent-Oriented Analysis and Design*, Autonomous Agents and Multi-Agent Systems, 3 (2000), pp. 285–312.

[221] D. WU, C. JENNINGS, J. TERPENNY, S. KUMARA, AND R. X. GAO, *Cloud-Based Parallel Machine Learning for Tool Wear Prediction*, Journal of Manufacturing Science and Engineering, Transactions of the ASME, (2018).

[222] XML, *XML. Extensible markup lan guage (XML) 1.0*.

[223] T.-H. YANG AND W.-P. LEE, *Intelligent Service Reconfiguration for Home Robots*, in Advances in Reconfigurable Mechanisms and Robots II, X. Ding, X. Kong, and S. J. Dai, eds., Springer International Publishing, Cham, 2016, pp. 735–745.

[224] A. M. A. YOUSSEF AND H. A. ELMARAGHY, *Optimal configuration selection for Reconfigurable Manufacturing Systems*, International Journal of Flexible Manufacturing Systems, 19 (2007), pp. 67–106.

[225] R. Y. ZHONG, C. XU, C. CHEN, AND G. Q. HUANG, *Big Data Analytics for Physical Internet-based intelligent manufacturing shop floors*, International Journal of Production Research, 55 (2017), pp. 2610–2621.

[226] J. ZHOU, D. DJURDJANOVIC, J. IVY, AND J. NI, *Integrated reconfiguration and age-based preventive maintenance decision making*, IIE Transactions, 39 (2007), pp. 1085–1102.

[227] A. ZOITL, *Real-Time Execution for IEC 61499*, International Society of Automation, 2009.

[228] P. ZOLTAN AND E. GEORGE, *Runtime Reconfiguration in Networked Embedded Systems*, Internet of Things, Springer Singapore, Singapore, 1 ed., 2016.