A Collaborative Approach to Fault Tolerance in Cyber-Physical Systems

Luis Fernando Piardi

Supervisor:

Pedro Luís Gomes Cerqueira da Costa, Ph.D., Universidade do Porto

André Schneider, Ph.D., Univers. Tecnológica Federal do Paraná

Paulo Leitão, Ph.D., Instituto Politécnico de Bragança

Tese de doutoramento em regime de cotutela entre a Universidade do Porto e a Universidade Tecnológica Federal do Paraná

Programa Doutoral em Engenharia Electrotécnica e de Computadores (FEUP)
&
Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial (UTFPR)

June 26, 2025

© Luis Fernando Piardi: June 26, 2025

Faculdade de Engenharia da Universidade do Porto
&
Universidade Tecnológica Federal do Paraná

A Collaborative Approach to Fault Tolerance in Cyber-Physical Systems

Luis Fernando Piardi

Programa Doutoral em Engenharia Eletrótecnica e de Computadores
&
Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial

Approved by:

President: Prof. Dr. André Schneider de Oliveira, Universidade Tecnológica Federal do Paraná

Referee: Prof. Dr. Antonio Paulo Gomes Moreira, Universidade do Porto

Referee: Prof. Dr. Marco Antonio Simões Teixeira, Pontifícia Universidade Católica do Paraná

Referee: Prof. Dr. Marco Aurélio Wehrmeister, Universidade Tecnológica Federal do Paraná

Referee: Prof. Dr. Ricardo José Rabelo, Universidade do Porto

Supervisor: Prof. Dr. Pedro Luís Gomes Cerqueira Costa, Universidade do Porto

Assinado por: **Pedro Luís Cerqueira Gomes da Costa**
Num. de Identificação: 10065441
Data: 2025.07.09 11:15:01+01'00'

June 26, 2025

To my family.

ACKNOWLEDGEMENTS

First, I would like to thank God.

This thesis is the result of years of research, but it also reflects a journey of learning and personal development built throughout my life. The knowledge and values gained during its development are lessons I hope to apply with wisdom, integrity, and justice, always striving to promote good and commit to doing what is right. I want to take this opportunity to express my gratitude to all those who contributed to making this thesis a reality. In particular, I would like to acknowledge some of them here.

I would like to express my deepest gratitude to my research supervisor, Professor Paulo Leitão, for trusting me during this Ph.D. journey. I am also grateful for his encouragement, advice, and insightful discussions, all of which greatly contributed to this thesis's development and the improvement of its contributions. I also sincerely thank Professor Pedro Costa and Professor André Schneider, whose guidance and support have been invaluable throughout this journey.

To the Universidade Tecnológica Federal do Paraná (UTFPR) and the Faculty of Engineering of the University of Porto (FEUP) for providing me with the opportunity to pursue this Ph.D. under a cotutelle agreement. Additionally, I would like to thank the Polytechnic Institute of Bragança, particularly the Research Centre in Digitalisation and Intelligent Robotics (CeDRI), for the opportunity to carry out this research work at their facilities.

I would like to thank you for the financial support to the Fundação para a Ciência e Tecnologia (FCT) within the scope of the Ph.D. Grant UI/BD/151286/2021.

I would like to thank Eduarda Nepomuceno, my fiancée, who played a fundamental role in this journey. Her belief in my capabilities, along with her unwavering support, care, and understanding throughout the development of this thesis, has been invaluable. I am look forward to continuing to grow and build my life alongside her.

I am also profoundly grateful to my family, who have driven my courage and determination to realize my goals. I hope that one day I can make up for the moments of absence and distance and that time will be generous enough to allow me to enjoy many quality moments with each of you. I would like to particularly recognize Chirlei Débora Pereira, my mother, who has been a model of perseverance and has taught me always to do my best with the resources available. I extend my heartfelt thanks to my siblings Joana Paula Piardi, Lucas Matheus Piardi,

Felipe Bohnen Piardi, and Artur Bohnen Piardi, as well as to my father, José Cesário Piardi; it is a privilege to share life with all of you. I would also like to thank Estela Maris Bohnen, an essential person who fosters integration within our family.

I am grateful to all the colleagues from CeDRI and UTFPR who, along the way, became friends. There are too many to name individually, but each one knows the part they played, and each one holds a special place in my memories. Thank you all for being part of this process.

Luis Fernando Piardi

Ninguém ignora tudo. Ninguém sabe tudo.
Todos nós sabemos alguma coisa. Todos nós
ignoramos alguma coisa. Por isso, aprendemos
sempre. (FREIRE, Paulo, 1996).

ABSTRACT

PIARDI, Luis. **A Collaborative Approach to Fault Tolerance in Cyber-Physical Systems**. 2025. 247 p. Thesis (Double PhD in Electrical Engineering and Industrial Informatics and Electrical and Computer Engineering) – Universidade Tecnológica Federal do Paraná and Faculdade de Engenharia da Universidade do Porto. Curitiba, 2025.

The reliability and robustness of cyber-physical systems are critical aspects of the current industrial landscape. The high level of autonomous and distributed components associated with many devices makes cyber-physical systems prone to faults. With this in mind and contextualized with the current era of Industry 4.0, fault tolerance is essential for maintaining the robustness and resilience of systems facing unforeseen or undesirable disturbances. Despite their importance and benefits, traditional approaches for fault tolerance, such as local or monolithic centralized, do not correspond with the accelerated technological evolution pace over the past two decades. Driven by the advent of digital technologies such as the Internet of Things, cloud and edge computing, and artificial intelligence associated with enhanced computational processing and communication capabilities, the traditional fault tolerance approaches are out of sync with contemporary and future systems. Consequently, these approaches are limited in achieving the maximum benefits enabled by integrating these technologies, such as robustness related to fault tolerance accuracy and performance improvements, maintaining the flexibility and scalability fundamental for cyber-physical systems operations. Accordingly, this thesis proposes a collaborative fault tolerance approach for cyber-physical systems called Collaborative Fault-Star (CF*), a unified term that refers to the collaboration on the detection, diagnosis, and recovery stages of fault tolerance. The proposed approach takes advantage of cyber-physical components' inherent data analysis and communication capabilities and is based on multi-agent system principles, where components are self-fault tolerant and adopt collaborative and distributed intelligence behavior when necessary to transcend their individual fault tolerance capabilities. This thesis introduces the backbone and infrastructure of CF*, highlighting the components, agent behavior, functionalities, and interaction protocols to explore the benefits of communication and collaboration between agents to enhance fault tolerance. The CF* is a generic approach, and can be instantiated for diverse cyber-physical system domains as well as for different applications. Experiments were conducted on a developed laboratory-scale warehouse, focusing on the fault detection and diagnosis stage for temperature and humidity sensors in the warehouse racks. The experimental results confirmed the accuracy and performance improvements under CF* compared with the local approach and competitiveness compared to a centralized approach.

Keywords: Cyber-physical System. Fault Tolerance. Multi-Agent System. Industry 4.0.

RESUMO

PIARDI, Luis. **Uma Abordagem Colaborativa para Tolerância a Falhas em Sistemas Ciber-Físicos**. 2025. 247 f. Tese (Doutorado em cotutela em Engenharia Elétrica e Informática Industrial e Engenharia Electrotécnica e de Computadores) – Universidade Tecnológica Federal do Paraná e Faculdade de Engenharia da Universidade do Porto. Curitiba, 2025.

A confiabilidade e a robustez dos sistemas ciber-físicos são aspectos críticos no cenário industrial atual. O elevado nível de autonomia e distribuição dos componentes, associado à ampla variedade de dispositivos, torna esses sistemas suscetíveis a falhas. Nesse contexto, e alinhado à era atual da Indústria 4.0, a tolerância a falhas torna-se essencial para manter a robustez e a resiliência dos sistemas diante de perturbações imprevistas ou indesejáveis. Apesar de sua importância e dos benefícios oferecidos, as abordagens tradicionais de tolerância a falhas, como as estratégias locais ou centralizadas monolíticas, não acompanham o ritmo acelerado da evolução tecnológica das últimas duas décadas. Impulsionadas pelo advento de tecnologias digitais, como a Internet das Coisas, computação em nuvem e *edge*, e inteligência artificial, associadas a capacidades elevadas de processamento computacional e comunicação, as abordagens tradicionais de tolerância a falhas mostram-se defasadas em relação aos sistemas contemporâneos e futuros. Como consequência, tais abordagens apresentam limitações para alcançar os benefícios máximos proporcionados pela integração dessas tecnologias, como maior robustez, precisão na tolerância a falhas e melhorias de desempenho, mantendo a flexibilidade e escalabilidade fundamentais para a operação dos sistemas ciber-físicos. Diante disso, esta tese propõe uma abordagem colaborativa de tolerância a falhas em sistemas ciber-físicos, denominada *Collaborative Fault-Star* (CF*), um termo unificado que engloba a colaboração nos estágios de detecção, diagnóstico e recuperação de falhas. A abordagem proposta explora as capacidades inerentes de análise de dados e comunicação dos componentes ciber-físicos, fundamentando-se nos princípios de sistemas multiagente, nos quais os componentes são autônomos para tolerar falhas e adotam comportamento colaborativo e inteligência distribuída quando necessário, a fim de superar as limitações de suas capacidades individuais. Esta tese apresenta a infraestrutura e a espinha dorsal do CF*, destacando os componentes, comportamentos dos agentes, funcionalidades e protocolos de interação, de modo a explorar os benefícios da comunicação e colaboração entre agentes para aprimorar a tolerância a falhas. O CF* constitui uma abordagem genérica, passível de instanciação para diferentes domínios e aplicações de sistemas ciber-físicos. Experimentos foram conduzidos num armazém em escala laboratorial desenvolvido, com foco nos estágios de detecção e diagnóstico de falhas em sensores de temperatura e umidade instalados nas estantes do armazém. Os resultados experimentais confirmaram melhorias em termos de precisão e desempenho da abordagem CF* em comparação com a abordagem local e competitividade em relação à abordagem centralizada.

Palavras-chave: Sistema Ciber-Físico. Tolerância a Falha. Sistema Multi-Agente. Industria 4.0.

LIST OF FIGURES

Figure 1 – Classical design science research methodology. 31
Figure 2 – Details of organization structure of the thesis. 32
Figure 3 – Cycle and relation between fault, error and failure. 35
Figure 4 – Concept relationship tree for the thesis. Added terms are in blue. 38
Figure 5 – Fault Dimensions. 41
Figure 6 – CPS with key technology enablers. 45
Figure 7 – IIRA functional domains, crosscutting functions and system characteristics. 49
Figure 8 – RAMI 4.0 reference architecture. 50
Figure 9 – The 5C CPS architecture with related applications and techniques. 51
Figure 10 – Evolution of published scientific articles focusing FT in CPS over the time. . 53
Figure 11 – FT attributes in CPS based on analysis of retrieved abstracts ranked by occurrence frequency. 54
Figure 12 – FT means and techniques based on analysis of retrieved abstracts ranked by occurrence frequency. 55
Figure 13 – Related technologies based on analysis of retrieved abstracts. 56
Figure 14 – Top CPS domains using FT based on analysis of retrieved abstracts ranked by occurrence frequency. 57
Figure 15 – Timeline of different approaches for fault tolerance in the context of cyber-physical systems. 58
Figure 16 – Representation and mains mechanisms of an agent. 65
Figure 17 – FT within CPS architecture implemented with MAS technology. 75
Figure 18 – Cyber-physical component for MAS-based FT in CPS. 77
Figure 19 – Agent interaction for FT in the proposed CPS architecture. 81
Figure 20 – Orchestration and concerns for deployment of MAS for CPS in cloud and edge continuum. 83
Figure 21 – Overview of collaborative process for MAS-based FT for CPS. 85
Figure 22 – Representation and description of the collaboration parameters in CF*. . . . 86
Figure 23 – CF* meta-model representing the internal modules of cyber physical components and the collaborative FT stages. 89
Figure 24 – Structure of the system model of $agent_i$. 91
Figure 25 – Description of collaborative fault detection module for $agent_i$. 92
Figure 26 – Description of collaborative fault diagnosis module for $agent_i$. 94
Figure 27 – Description of collaborative fault recovery module for $agent_i$. 98
Figure 28 – Protocol for sharing information of new agent launched. 103
Figure 29 – Agent Disconnection protocol. 104
Figure 30 – Collaboration fault detection protocol. 105
Figure 31 – Collaboration fault diagnosis protocol. 106
Figure 32 – Protocol to mitigate the fault impact. 107
Figure 33 – Protocol for collaborative recovery action plan elaboration. 108
Figure 34 – Protocol for collaborative recovery execution. 109
Figure 35 – Algorithm of the 7CM to evaluate the collaboration for FDD. 111
Figure 36 – Example of transition explosion in the graphical Petri net formalism. 120
Figure 37 – Fundamental agent behavior modeled via PN integrating CF* approach. . . 122
Figure 38 – Excerpt of the fundamental PN referring to the control thread. 123

Figure 39 – Excerpt of the fundamental PN referring to the interaction thread. 124
Figure 40 – Excerpt of the fundamental PN referring to the fault tolerance thread. 125
Figure 41 – Excerpt of the fundamental PN referring to the models improvement thread. 126
Figure 42 – PN refinement modeling the dynamic of collaborative fault detection. 131
Figure 43 – PN refinement modeling the dynamic of collaborative fault diagnosis. 133
Figure 44 – PN refinement modeling the dynamic of collaborative fault recovery. 136
Figure 45 – Structural and behavioral analysis of the PN of the agent with CF* approach. 139
Figure 46 – Gantt diagram for performance analysis of the fundamental PN for scenario #1.144
Figure 47 – Evolution of tokens in places of fundamental PN for scenario #1. 144
Figure 48 – Gantt diagram for performance analysis of the fundamental PN for scenario #2.145
Figure 49 – Evolution of tokens in places of fundamental PN for scenario #2. 146
Figure 50 – Gantt diagram for performance analysis of the fundamental PN for scenario #3.147
Figure 51 – Evolution of tokens in places of fundamental PN for scenario #3. 147
Figure 52 – Warehouse layout floor reproduced in the case study. 151
Figure 53 – Flow of products inside the warehouse for the case study. 152
Figure 54 – ARENA workbench representing the case study in laboratory scale. 153
Figure 55 – Overview of the CPS for ARENA case study. 154
Figure 56 – Description of the ARENA cyber-physical components in ROS environment. 155
Figure 57 – Five cyber-physical components racks used in the detection stage experiments.158
Figure 58 – Dataset used for training the fault detection models for each agent. 159
Figure 59 – Dataset with fault-free and faulty samples used in fault detection experiments. 160
Figure 60 – Procedure used for training the SVM models for detection experiments. . . 161
Figure 61 – Overview of the experimental procedure developed for fault detection experiments. 162
Figure 62 – Comparison between the MCFT, LFT, and CF* for fault detection experiment with fault-free samples. 164
Figure 63 – Comparison between the MCFT, LFT, and CF* for fault detection experiment with faulty samples. 165
Figure 64 – Data test used in each agent for the three experiments related to 7CM. . . . 167
Figure 65 – Individual agents performance for fault detection based on data test for 7CM experiment. 168
Figure 66 – Performance of agents with collaborative fault detection before the adjust by 7CM. 169
Figure 67 – Performance of agents after the adjustment of the collaborations parameters by 7CM analysis. 171
Figure 68 – Data test for diagnosis experiments with faults highlighted in the gray area. . 175
Figure 69 – Proposed procedure to compare the LFT and CF* for the fault diagnosis stage.176
Figure 70 – Box plot comparing the diagnosis experiments based on LFT with CF* with data test 1. 179
Figure 71 – Box plot comparing the diagnosis experiments based on LFT with CF* with data test 2. 180
Figure 72 – Comparison of confusion matrix of agent $agent_8$ using SVM model with diagnosis data test 2 180
Figure 73 – Best student paper award received in IEEE ICIT 2022 conference. 214
Figure 74 – Highly commended paper award received in ROBOT2019 conference. . . . 215
Figure 75 – Flowchart of internal operation for Collaborative Fault Detection of $agent_i$. 218
Figure 76 – Flowchart of internal operation for Collaborative Fault Diagnosis of $agent_i$. 220
Figure 77 – Flowchart of internal operation for Collaborative Fault Recovery of $agent_i$. . 222

Figure 78 – Fundamental concepts of ARENA. 224
Figure 79 – Exploded view of the ARENA's infrastructure. 226
Figure 80 – Rack infrastructure overview. 227
Figure 81 – WsBot description and components. 228
Figure 82 – WsBot electronic schematic. 229
Figure 83 – Virtual LiDAR overview. 232
Figure 84 – Top view of the ARENA with AR element. 233
Figure 85 – Interaction for storage negotiation in the ARENA warehouse. 234
Figure 86 – Interaction for transport negotiation in the ARENA warehouse. 235
Figure 87 – Response rate of the centralized vs decentralized system by increasing the number of agent robots. 237
Figure 88 – Experiments setup to analyze the behavior of the MAS based CPS when increasing products agents. 238
Figure 89 – Transporting time when increasing products agents. 239
Figure 90 – Experiments setup to analyze the behavior of the system when increasing robot agents. 239
Figure 91 – Responsiveness and flexibility analysis when increasing products agents. . . 240
Figure 92 – Divisibility of the robot agents to carry out the transports in the warehouse. . 243
Figure 93 – Dataset used for $agent_1$ to training diagnosis models. 245
Figure 94 – Dataset used for $agent_2$ to training diagnosis models. 245
Figure 95 – Dataset used for $agent_3$ to training diagnosis models. 245
Figure 96 – Dataset used for $agent_4$ to training diagnosis models. 245
Figure 97 – Dataset used for $agent_5$ to training diagnosis models. 246
Figure 98 – Dataset used for $agent_6$ to training diagnosis models. 246
Figure 99 – Dataset used for $agent_7$ to training diagnosis models. 246
Figure 100 –Dataset used for $agent_8$ to training diagnosis models. 246
Figure 101 –Dataset used for $agent_9$ to training diagnosis models. 246
Figure 102 –Dataset used for $agent_{10}$ to training diagnosis models. 247
Figure 103 –Dataset used for $agent_{11}$ to training diagnosis models. 247
Figure 104 –Dataset used for $agent_{12}$ to training diagnosis models. 247

LIST OF TABLES

Table 1 – CPS application domains highlighting their features and related works. . . . 47
Table 2 – Categories of FT methods for CPS. 62
Table 3 – Summary of studies with FT approaches, methods, and techniques in the context of CPS. 63
Table 4 – The main research challenges to develop FT in CPS. 70
Table 5 – Examples of instantiation strategies for the collaborations parameters covering all FT stages. 88
Table 6 – Collaborative fault detection stage features. 94
Table 7 – Collaborative fault diagnosis stage features. 97
Table 8 – Collaborative fault recovery stage features. 101
Table 9 – Summary of the 7CM to evaluate collaborative fault detection and diagnosis in CPS. 114
Table 10 – Summary of the structural properties of the fundamental PN for an agent with CF* approach. 140
Table 11 – Summary of the behavioral properties of the fundamental PN for an agent with CF* approach. 141
Table 12 – Ticks associated with the transitions for the performance analysis. 143
Table 13 – Cyber-physical components in the ARENA case study. 156
Table 14 – Parameters used in the SVM model for the fault detection experiment. 161
Table 15 – Instantiation strategies for the collaboration parameters of CF* in fault detection experiments. 162
Table 16 – Accuracy for fault detection experiment with fault-free data test. 163
Table 17 – Accuracy for fault detection experiment with precision degradation fault data test. 166
Table 18 – Result of 7CM for experiment with collaboration for fault detection. 169
Table 19 – Training and test dataset for diagnosis of DHT22 sensor. 174
Table 20 – Instantiation strategies for the collaboration parameters of CF* in fault diagnosis experiments. 177
Table 21 – Result for diagnosis in terms of F1-Score (%) for data test 1. 178
Table 22 – Result for diagnosis in terms of F1-Score (%) for data test 2. 179
Table 23 – Result of Wilcoxon Signed-Rank for Paired Samples test. 181
Table 24 – Mechanical characteristics of WsBot. 228
Table 25 – Statistical results of the total of 9 experiments carried out for transport time (R indicate robots and P indicate product). 240

LIST OF ACRONYMS

ACRONYMS

7CM	Seven Collaboration Metrics
ACL	Agent Communication Language
AI	Artificial Intelligence
AMR	Autonomous Mobile Robot
AR	Augmented Reality
AUML	Agent Unified Modeling Language
CF*	Collaborative Fault-Star
CFP	Call for Proposal
CNC	Computer Numeric Control
CoAP	Constrained Application Protocol
CPS	Cyber-Physical Systems
DAI	Distributed Artificial Intelligence
DC	Direct Current
DFT	Decentralized Fault Tolerance
DT	Digital Twin
FDD	Fault Detection and Diagnosis
FIFO	First in first out
FIPA	Foundation for Intelligent Physical Agents
FT	Fault Tolerance
GNB	Gaussian Naive Bayes
HMI	Human Machine Interface
I2C	Inter-Integrated Circuit
I4.0	Industry 4.0
ICT	Information and Communications Technology
IIRA	Industrial Internet Reference Architecture
IoT	Internet of Things
JADE	Java Agent DEvelopment Framework
KNN	K-Nearest Neighbors
LFT	Local Fault Tolerance
LPWAN	Low-Power Wide-Area Networks
M2M	Machine to Machine
MAS	Multi-Agent Systems
MCFT	Monolithic Centralized Fault Tolerance

ML	Machine Learning
MLP	Multi-Layer Perceptron
MQTT	Message Queuing Telemetry Transport
NIST	National Institute of Standards and Technology
NLTK	Natural Language ToolKit
NuSVM	Nu-Support Vector Machine
OPC UA	Open Platform Communications Unified Architecture
PID	Proportional Integral Derivative
PN	Petri net
PndK	Petri nets development toolKit
QoS	Quality of Service
RAMI 4.0	Reference Architecture Model for Industrie 4.0
RF	Random Forest
ROS	Robot Operating System
RSSI	Received Signal Strength Indication
SDN	Software-Defined Network
SLAM	Simultaneous Localization and Mapping
SPI	Serial Peripheral Interface
SVM	Support Vector Machine
UML	Unified Modeling Language
VDI	Association of German Engineer

LIST OF NOMENCLATURE AND SYMBOLS

INDEXES

i	Index of agent, designated as local or initiator agent.
j	Index of agent, designated as the external or collaborative or participant agent.
n	Number of components/agents existing in the system.
k	Number of predetermined fault classes that the diagnostic model can classify for the agent.
x	Index of fault class.

SYMBOLS

$agent_i$	Element of set $\mathcal{A}$ (used as local or initiator agent).
$agent_j$	Element of set $\mathcal{A}$ (used as external or participant agent).
ϵ_i	Raw data of $aegnt_i$.
φ_i	Data of ϵ_i with fault associated.

COMPONENTS SETS

$\mathcal{A}$	Set of agents.
MO_i	Set of Modules of $agent_i$.
CL_i	Set of Collaboration Protocols of $agent_i$
PC_i	Set of Physical Component associated to the $agent_i$.
Ac_i	Set of actuators of PC_i.
Se_i	Set of sensors of PC_i.

MATHEMATICAL SETS

$\mathbb{N}$	Set of natural numbers.
$\mathbb{N}^*$	Set of natural numbers greater than zero.

FAULT TOLERANCE SYMBOLS

Fdt_i	Function or model for fault detection stage of $agent_i$.
Dt_i	Output of fault detection stage of $agent_i$.
pdt_i	Probability or confidence of fault detection output of $agent_i$.
CDt_i	Collaboration model for fault detection stage of $agent_i$.
Fdi_i	Function or model for fault diagnosis stage of $agent_i$.
Di_i	Output of fault diagnosis stage of $agent_i$.
Di_i^*	Output of fault diagnosis of $agent_i$ shared with external agents.
CDi_i	Collaboration model for fault diagnosis stage of $agent_i$.

$\mathcal{H}_i$	Set with history of mapped failures used for correlation in the diagnosis stage of $agent_i$.
hf	Element of set $\mathcal{H}$.
$\mathcal{K}_i$	Set of fault classes of $agent_i$.
f_x	Element of set $\mathcal{K}$.
$\mathcal{SI}$	Set of similarity of diagnosed faults.
sf	Element of set $\mathcal{SI}$.
$\mathcal{CA}$	Set of possible causes for the fault f diagnosed.
$\mathcal{IM}$	Set of possible impacts for the fault f diagnosed.

Fr_i	Function or model for fault recovery stage of $agent_i$.
R_i	Output of fault recovery stage of $agent_i$.
CDt_i	Collaboration model for fault recovery stage of $agent_i$.
re_i	Actions for local fault recovery of $agent_i$.
$\mathcal{RE}_i$	Set of actions to recover from a global failure proposed by $agent_i$.
re_{ij}	Fault recovery action element of set $\mathcal{RE}_i$ recommended by $agent_i$ for $agent_j$.

COLLABORATION PARAMETERS

τ	Trigger parameter for collaboration.
σ	Selection Parameter for collaboration.
ρ	Decision-make Parameter for collaboration.

COLLABORATIONS METRICS SYMBOLS

qc	Quantity of collaboration.
qi	Quantity of inferences.
ri	Collaboration decision remain incorrect.
rc	Collaboration decision remain correct.
mi	Collaboration incorrectly modify the decision.
mc	Collaboration correctly modify the decision.
CM_1	Collaboration Metric 1 to evaluate the anxiety.
CM_2	Collaboration Metric 2 to evaluate the collective error.
CM_3	Collaboration Metric 3 to evaluate the insecurity.
CM_4	Collaboration Metric 4 to evaluate the error induction.
CM_5	Collaboration Metric 5 to evaluate the correction.
CM_6	Collaboration Metric 6 to evaluate the collaboration undesired.
CM_7	Collaboration Metric 7 to evaluate the collaboration desired.

PETRI NET SYMBOLS

P	Set of places.
T	Set of Transitions.
I	Set of input arcs in PN transitions.
O	Set of output arcs in PN transitions.

m_0	Initial positions of tokens in a PN.
t	Transition of the PN.
p	Place of the PN.

EXPERIMENT DATASET SYMBOLS

t	Time associated to the dataset sample
$h(t)$	Sensor data without noise or fault.
η	Noise with Gauss distribution with zero mean and low variance.
ν	Noise with Gauss distribution with zero mean and high variance.
$s(t)$	Sensor output without fault.
$s_{precision}(t)$	Sensor output with precision fault.
$s_{drift}(t)$	Sensor output with drift fault.
$b(t)$	Drift increment value added to the sensor output.
$s_{bias}(t)$	Sensor output with bias fault.

SUMMARY

1 INTRODUCTION 22
1.1 CONTEXT AND MOTIVATION 23
1.2 PROBLEM STATEMENT 25
1.3 THESIS RESEARCH SCOPE 26
1.3.1 Objective 26
1.3.2 Research Questions and Hypotheses 27
1.4 THESIS BOUNDARIES 28
1.4.1 Assumptions 29
1.4.2 Boundaries 29
1.5 RESEARCH METHODOLOGY 30
1.6 DOCUMENT STRUCTURE 32

2 RELATED WORK 34
2.1 FAULT TOLERANCE BACKGROUND 35
2.1.1 Taxonomy 37
2.1.2 Fault Dimensions 40
2.2 FUNDAMENTALS OF CYBER-PHYSICAL SYSTEMS 42
2.2.1 Definition and Components of CPS 43
2.2.2 Key Enabling Technologies for Establishment of CPS 45
2.2.3 Applications Domain of CPS 47
2.2.4 Reference Architectures for Design CPS in Industrial Context 48
2.3 FAULT TOLERANCE IN CYBER PHYSICAL SYSTEMS 51
2.3.1 Literature Characterization 52
2.3.2 Related Approaches 57
2.3.3 Principal Methods 62
2.4 MULTI-AGENT SYSTEMS TO SUPPORT FT IN CPS 64
2.4.1 Multi-Agent Systems Concept and Applications 64
2.4.2 Why Multi-Agent Systems to Address FT in CPS? 67
2.5 CHALLENGES AND RESEARCH OPPORTUNITIES 69
2.5.1 Broad Perspective 69
2.5.2 Research Gaps Addressed 72
2.6 SUMMARY 72

3 COLLABORATIVE FAULT TOLERANCE FOR CYBER-PHYSICAL SYSTEMS 74
3.1 MAS-BASED FAULT TOLERANCE ARCHITECTURE FOR CYBER-PHYSICAL SYSTEMS 75
3.1.1 Cyber-Physical Component and Cyber-Physical Interaction 76
3.1.2 Cyber-to-Cyber Interaction 80
3.1.3 Edge Cloud Continuum for Agent Deployment 82
3.2 MAS-BASED COLLABORATIVE FAULT TOLERANCE 84
3.2.1 The CF* Approach 84
3.2.2 Collaborative Parameters for the CF* 86
3.3 CF* META-MODEL FOR A GENERIC AGENT 87
3.4 MODULES OF CF* FAULT TOLERANCE LAYER 90
3.4.1 System Model 90

3.4.2 Collaborative Fault Detection . . . 92
3.4.3 Collaborative Fault Diagnosis . . . 94
3.4.4 Collaborative Fault Recovery . . . 97
3.5 AGENT INTERACTION FOR FT BASED ON CF* . . . 101
3.5.1 Interaction for Agent Information . . . 102
3.5.2 Interaction for Collaborative Fault Detection . . . 104
3.5.3 Interaction for Collaborative Fault Diagnosis . . . 105
3.5.4 Interaction for Collaborative Fault Recovery . . . 107
3.6 EVALUATION METRICS FOR COLLABORATION . . . 110
3.7 SUMMARY . . . 114

4 AGENT MODEL AND DYNAMIC BEHAVIOR SPECIFICATION FOR CF* . 116
4.1 PETRI NETS TO MODEL THE AGENT DYNAMIC BEHAVIOR . . . 117
4.2 AGENT DYNAMIC MODEL WITH CF* BEHAVIOR . . . 120
4.2.1 Fundamental Agent Behavior Modeled by Petri Net . . . 121
4.2.2 Control Thread . . . 123
4.2.3 Interaction Thread . . . 124
4.2.4 Fault Tolerance Thread . . . 125
4.2.5 Models Improvement Thread . . . 126
4.2.6 Mathematical Modeling of the Fundamental Agent Behavior PN . . . 127
4.3 PETRI NETS REFINEMENT OF THE FAULT TOLERANCE BEHAVIOR 129
4.3.1 Refinement of Collaborative Fault Detection Transition . . . 130
4.3.2 Refinement of Collaborative Fault Diagnosis Transition . . . 132
4.3.3 Refinement of Collaborative Fault Recovery Transition . . . 135
4.4 FORMAL ANALYSIS OF FUNDAMENTAL PETRI NET OF CF* . . . 138
4.4.1 Structural Analysis . . . 138
4.4.2 Behavioral Analysis . . . 141
4.4.3 Performance Analysis . . . 142
4.5 SUMMARY . . . 148

5 EXPERIMENTS AND RESULTS . . . 149
5.1 CASE STUDY TO PROOF OF CONCEPT . . . 150
5.1.1 Case Study Description . . . 151
5.1.2 ARENA Workbench Development and Components . . . 153
5.1.3 How the ARENA Case Study was Used for the CF* Proof of Concept . 156
5.2 EXPERIMENTS FOR FAULT DETECTION IN THE RACK SENSOR . 157
5.2.1 Dataset Elaboration for Fault Detection Experiments . . . 158
5.2.2 Experimental Procedure . . . 160
5.2.3 Results and Discussion for Detection Stage . . . 162
5.3 ADOPTING THE 7CM TO ENHANCE THE CF* PARAMETERS . . . 166
5.3.1 Data Test Used for Experiments . . . 166
5.3.2 Experiments Evaluation and Results Adopting 7CM . . . 167
5.3.3 Discussion related to the 7CM . . . 171
5.4 EXPERIMENTS FOR FAULT DIAGNOSIS IN THE RACK SENSOR . . 172
5.4.1 Dataset Elaboration for Fault Diagnosis Experiment . . . 173
5.4.2 Experimental Procedure . . . 174
5.4.3 Evaluation Metrics . . . 177
5.4.4 Results for Diagnosis Stage . . . 178
5.4.5 Wilcoxon Test . . . 181

5.5 CONJECTURE FOR RECOVERY STAGE FOR FAULTS IN THE RACK SENSOR . . . 182
5.6 SUMMARY . . . 183

6 CONCLUSION AND FUTURE WORKS . . . 185
6.1 MAIN CONTRIBUTIONS . . . 186
6.2 LIMITATIONS AND DISCUSSION . . . 188
6.3 FUTURE WORK . . . 189

REFERENCES . . . 192

APPENDIX 211

APPENDIX A PUBLISHED PAPERS AND AWARDS . . . 212

APPENDIX B MODULES OF CF* CONTROL LOOP LAYER . . . 216

APPENDIX C FLOWCHARTS TO DESCRIBE THE CF* APPROACH . . . 218
C.1 FLOWCHART OF COLLABORATIVE FAULT DETECTION . . . 218
C.2 FLOWCHART OF COLLABORATIVE FAULT DIAGNOSIS . . . 220
C.3 FLOWCHART OF COLLABORATIVE FAULT RECOVERY . . . 222

APPENDIX D DETAILS OF ARENA AND EXPERIMENTS FOR WAREHOUSE OPERATION . . . 224
D.1 ARENA PHYSICAL STRUCTURE . . . 225
D.2 RACKS INFRASTRUCTURE . . . 226
D.3 THE TINY AND LOW-COST MOBILE ROBOT: WSBOT . . . 227
D.4 SENSORS DEDICATED TO WSBOT IN ARENA . . . 230
D.4.1 Localization System . . . 230
D.4.2 Virtual LiDAR . . . 231
D.5 AUGMENTED REALITY FEATURE . . . 232
D.6 INTERACTION PROTOCOLS TO ARENA WAREHOUSE OPERATION 234
D.7 EXPERIMENTAL ANALYSIS FOR ARENA LOGISTIC OPERATION . 235
D.7.1 Scalability Analysis . . . 236
D.7.2 Transporting Time Analysis . . . 237
D.7.3 Transport Time Experiment varying the number of product agents . . . 240
D.7.4 Transport Time Experiment varing the number of robots agents . . . 242
D.7.5 Resource Usage or Divisibility . . . 243
D.8 QUALITATIVE DISCUSSION . . . 244

APPENDIX E TRAINING DATASET FOR DIAGNOSIS EXPERIMENT . . . 245

1 INTRODUCTION

The communication, information exchange, and collaboration among individuals of a species (e.g., humans or ants) are fundamental to enhancing knowledge, resilience in crises, and problem-solving to ensure species survival. By combining the skills and knowledge of different individuals, this approach enables a more efficient resolution of complex problems with a wider set of information. An analogy between these species and digital systems can be established, where communication, information exchange, and collaboration transcend the individual capacity of intelligent and autonomous components, i.e., Cyber-Physical Systems (CPS) element that integrates physical processes with computational capabilities, enabling sensing, control, data processing and communication. Thus, in addition to performing their functions autonomously, the component's interaction, data exchange, and information translates into continuous improvement, adapting to different operating conditions culminating in collective intelligence.

To support the digitization, distribution, communication, and control of physical components, as well as the implementation of new technologies, CPS is emerging as a key approach. CPS integrates computer systems with physical processes, enabling devices to exchange data, negotiate, and collaborate. This integration allows the system automation in terms of control, reconfiguration, and condition monitoring to maximizing the production efficiency through the data analysis and processing.

There is an increasing demand for reliability and robustness in CPS, as their components are constantly susceptible to anomalies and fault episodes. Detecting and diagnosing potential faults at the earliest opportunity is crucial, as well as implementing fault tolerant operations to minimize performance degradation and prevent hazardous conditions. The CPS is considered fault tolerant if it can continue to operate in presence of fault event, albeit with perhaps degraded performance. A CPS is deemed fault tolerant when the components and systems is able to sustain functional operations during a fault occurrence, albeit at a possibly reduced level of performance. Then, an appropriate Fault Tolerance (FT) approach is essential for maintaining component and system operation during fault episodes.

The remainder of this Chapter is organized as follows:

- Section 1.1: briefly contextualizes and discusses the fault in context of CPS.
- Section 1.2: introduce the problem of FT in CPS.

- Section 1.3: define the thesis objective, research questions and hypotheses.
- Section 1.4: defines the thesis assumptions, as well as its limitations.
- Section 1.5: presents the adopted research methodology used to guide the development of this thesis.
- Section 1.6: presents the complete organization of this document.

1.1 CONTEXT AND MOTIVATION

According to Edward A. Lee, CPS are defined as: *"Integrations of computation with physical processes. Embedded computers and networks monitor and control the physical processes, usually with feedback loops where physical processes affect computations and vice versa"* (LEE, 2008). In essence, CPS result from the integration of computing technologies, networking infrastructures, and real-world systems, with the goal of monitoring, controlling, and operating physical processes.

Another crucial factor in establishing CPS as a model for integrating cyber and physical components was the advent of Industry 4.0 (I4.0), emerged in Germany and promoted by Association of German Engineer (VDI) and the German government through projects funding, such as that by Kagermann *et al.* (2013). This industrial revolution introduced a new generation of production systems based on the concept of the smart factory. A smart factory can autonomously decide on actions in various scenarios, from equipment maintenance to intelligent and autonomous decision-making for each asset and product, thus meeting individual customer needs. The most effective way to develop such a factory is by decentralizing the coordination of functions and decision-making. Additionally, I4.0 is designed according to several technological principles that guide their implementation, including decentralized and autonomous control, flexibility, service orientation, connectivity, interoperability, optimization, real-time decision-making, virtualization, and human integration (KAGERMANN *et al.*, 2013). This shift moves from a centralized control system or local devices to intelligent components equipped with sensing, processing, and network communication capabilities (MAKANDA *et al.*, 2023).

CPS can act as the backbone infrastructure, creating a network of components with cyber and physical counterparts capable of making autonomous decisions in a decentralized manner, thereby implementing I4.0-compliant solutions, integrating digital processes and disruptive technologies, as for example, Artificial Intelligence (AI), cloud and edge computing,

Internet of Things (IoT), and big data (LEITÃO *et al.*, 2016). Therefore, CPS is one of the pillars facilitating the transition from traditional, monolithic centralized systems with isolated components disconnected, to decentralized and self-organized systems that provide agility, responsiveness, and adaptation to changing conditions and decision-making, thus enabling energy efficiency and cost-effectiveness (BARBOSA *et al.*, 2016; KARNOUSKOS *et al.*, 2012). In this context, CPS fosters the implementation of large-scale systems in various fields, including power generation (ZHANG *et al.*, 2017; MAFFEI *et al.*, 2017), water distribution (HOUSH; OHAR, 2018), industrial and manufacturing systems (LEE *et al.*, 2015), and other sectors to enhance production systems and improve human life.

The capabilities of CPS systems are promising in the perspective of substantial long-term, social, economic, and environmental benefits (ZHANG *et al.*, 2022). However, it is essential to take into account that both the cyber layer, the communication networks between components, and especially the physical layer are not completely deterministic or predictable and are susceptible to faults. The CPS, in its essence, operates in an uncontrolled environment, i.e., the physical world. Undesirable events, disturbances, and unexpected conditions are intrinsic to applications involving machines, actuators, sensors, network infrastructure, and algorithms, even in systems with rigorous design criteria (LEE, 2008).

CPS can be affected by both cyber and physical faults. The cyber environment can be affected by logical faults (e.g., processing problems), communication (e.g., network disconnection), and project design (e.g., software bugs). The physical environment is susceptible to various types of faults, including, for example, anomalies arising from sensor calibration or damage, manufacturing defects, actuator degradation, mechanical faults, performance degradation or electrical discontinuities (JAN *et al.*, 2021). Since a fault-free CPS is utopian, especially due to unexpected conditions in real environments, the process of equipping components with the ability to detect, diagnose and, as far as possible, recover from faults has been a topic of relevant interest in the academic and industry communities in recent years.

CPS-based applications, along with distributed and integrated technologies, are increasingly popular. However, these solutions often rely on traditional FT approaches, using local or centralized architectures, despite incorporating new algorithms and AI-driven data analysis models for fault detection, diagnosis, and recovery. Efficient FT for CPS requires more than advanced mathematical or statistical models due to their inherent complexity, both structurally and behaviorally (XU *et al.*, 2020).

Fault tolerance in CPS has not evolved at the same pace as the technology that integrates this type of system. Faults in the CPS may lead to the malfunction of the entire system and some catastrophic consequences. In order to meet the robustness and reliability demands, CPS should possess the ability of FT in an innovative and intelligent way, taking into account the distributed nature as well as the communication capacity through components interaction. Given these factors, the existing strategies for FT must be improved and specified, taking into account the characteristics associated with CPS, integrating new technologies such as AI and IoT, as well as the ability to exchange information between devices, enabling the collaboration for detection, diagnosis and recovery, improving reliability and robustness, without compromising the scalability and flexibility of the CPS.

1.2 PROBLEM STATEMENT

The FT approaches are typically deployed locally, i.e., each component locally integrates FT techniques, or is monolithic centralized, i.e., a central node is responsible for the fault tolerance of the CPS. Such practices are out of sync with the current technological reality, and even with the paradigm addressed by the CPS, which is based on distribution, decentralization, autonomy, and collaboration. Although CPS adopt technologies that emerged with the I4.0, implementing FT in agents locally or centrally introduces a single fault point and is consequently limited in achieving the maximum benefits enabled by the integration of these technologies, such as improvements in accuracy, performance, and flexibility (ZHOU *et al.*, 2015; XU *et al.*, 2018b).

Both centralized and local FT approaches have significant drawbacks. For the centralized approach, the main issues are: (i) scalability challenges, as an increase in devices leads to congestion and results in computational and communication overhead, potentially causing data loss and reduced responsiveness in fault detection and diagnosis; (ii) incompatibility with heterogeneous CPS components, as different FT models must be implemented on a single node; and (iii) vulnerability to severe faults, as the entire system depends on a single software component, creating a single point of failure. For the local approach, the main drawbacks are: (i) difficulty in managing global or generalized faults, where faults of the same nature are detected, diagnosed, and reported by multiple system components; and (ii) challenges in adapting to changing operating conditions (MORTELLEC *et al.*, 2013; PIARDI *et al.*, 2023).

Faced with the problem presented, some studies explore hierarchical or distributed FT

approaches (these will be discussed in detail in Chapter 2 of this thesis). However, most of these studies focus on specific or dedicated application of CPS. Additionally, existing distributed FT approaches do not fully leverage inherent CPS properties such as autonomy, communication, and collaboration. This indicates that current solutions for tolerating faults in distributed CPS are immature and lack a generic approach, i.e., can be adapted to different CPS applications. Such approach should incorporate scalability, autonomy, and collaboration to effectively detect, diagnose, and recover from disturbances or faults.

This justifies the interest in understanding fault tolerance competencies in distributed CPS. By integrating principles of communication and collaboration, the autonomous capabilities of CPS components to detect, diagnose, and recover from faults can be enhanced. Thus, significant contributions can be made to the scientific community in this field.

1.3 THESIS RESEARCH SCOPE

The research presented in this thesis focuses on proposing an approach suitable for collaborative fault tolerance in CPS.

1.3.1 Objective

Considering the importance of FT in CPS and motivated by the problem discussed in the previous Section, this thesis aims to address the challenges regarding the design and development of a distributed approach for FT in CPS. This work proposes, specifies, and formalizes a collaborative approach for fault tolerance, entitled Collaborative Fault-Star (CF*), which introduces the self-fault tolerance capability in cyber components and adopts collaborative and distributed behavior to improve the asset FT capability. The proposed CF* is based on Multi-Agent Systems (MAS) concepts and principles, and the FT is organized into stages, namely, collaborative fault detection, collaborative fault diagnosis, and collaborative fault recovery, representing the behaviors implemented in each component of CPS. These stages take advantage of the CPS's ability to analyze data and component communication to exchange information between the same or between different stages to improve the FT capabilities in the CPS compared to traditional FT approaches, i.e., local or monolithic centralized.

The proposed approach addresses collaboration between different cyber-physical components for fault detection, diagnosis, and recovery by parameterizing three main behaviors:

when to collaborate, with which components to collaborate, and how to make-decision based on collaboration outcome. Based on the MAS principles, such as autonomy, intelligence and interaction, the components can collaborate in four ways: detection support, diagnosis support, sharing diagnosis information, and recovery support. Additionally, the proposed approach is suitable for deployment in diverse CPS applications and domains, supporting different AI-based methods to execute the fault tolerance stages.

With this in mind, this work postulates that the enhancement of FT in CPS can be achieved by incorporating collaboration between cyber physical components, adopting a distributed middleware, based on a MAS. The preference for the MAS paradigm is due to its inherent agent features of intelligence, autonomy, data analysis, and interaction between agents. Additionally, the MAS paradigm is a widespread approach based on distributed architectures (LI *et al.*, 2016), making it suitable for designing and developing intelligent distributed systems that require modularity, flexibility, adaptability, scalability and collaboration (QUEIROZ *et al.*, 2021). Agent-based systems enable the creation of models with autonomy (based on intelligent algorithms) and interaction (based on interaction protocols) between components, facilitating decisions, information sharing, and interaction with other components essential for the CPS (KARNOUSKOS *et al.*, 2020).

1.3.2 Research Questions and Hypotheses

Based on the problem statement presented in Section 1.2 and the literature review conducted during the development of this thesis, the research gaps were identified regarding FT in CPS. These gaps were translated into research questions aimed at addressing the objective of this thesis, i.e., to conceptualize an approach for FT that integrates concepts of collaboration and interaction between components for CPS. Accordingly, the research questions were defined to guide the study conducted in this doctorate and described in this document:

- ***RQ-1:*** *What requirements and features must be considered to conceptualize an approach that serves as a generic model for implementing FT in CPS addressing robustness, and scalability requirements without compromising system dependability?*
- ***H-1***: *If CPS benefit from autonomy, intelligence, interaction and collaboration of components at the operational and control levels, then incorporating features (supported for example by MAS, AI and IoT) and collaboration between components for FT can enhance*

detection, diagnosis, and recovery capability. This integration can improve the resilience and robustness of the CPS preserving system scalability and reliability.

The approach conceptualized by this thesis, based on reflections obtained from ***RQ-1***, should contain aspects, requirements and features aligned with the current technological scenario, as well as the interaction and collaboration between cyber-physical components to transcend the individual competence for fault detection, diagnosis and recovery. Furthermore, the approach should be versatile for deployment across diverse fields as referenced in CPS.

The ***RQ-2*** serves as a critical guide throughout the entirety of the thesis development. This research question not only helps to elucidate the capabilities inherent within the proposed collaborative approach but also plays a pivotal role in identifying and defining its inherent limitations. Consequently, it ensures a comprehensive understanding and provides clearer investigation direction, contributing significantly to the thesis research contributions.

- ***RQ-2:*** *Under which circumstances does collaboration for FT between distinct components outperform traditional approaches?*

- ***H-2***: *Collaboration for fault tolerance between distinct components outperforms traditional approaches in circumstances where systems face complex, interdependent, and dynamic environments by reducing single points of faults and leveraging the individual experiences and specialized competencies of different agents to fault detect, diagnoses and recovery process. Through collaborative efforts enhances the agent resilience and also improves the robustness of fault detection, diagnosis, and recovery through allowing the agent to capitalize on diverse insights, information and capabilities unattainable by individual agents, thereby outperforming traditional capacity capabilities in complex, interdependent environments.*

1.4 THESIS BOUNDARIES

In order to fulfill the proposed objectives, and based on the aspects discussed in the previous Section the following assumptions are taken into account when defining the hypothesis ***H-1***, ***H-2*** and formulating the research questions to be explored in this thesis.

1.4.1 Assumptions

Considering the broad spectrum of potential applications of the proposed approach, it is essential to precisely delineate and restrict the scope and boundaries of the proposed hypothesis.

- CPS are widely adopted in the context of digitization, automation, and industrial architecture, where the services provided by these systems require robustness, resilience, and a high degree of availability and dependability.
- The agents of the MAS are self-contained. They are autonomous entities within the CPS that can operate and control assets independently without external intervention. These agents handle all tasks related to the asset, from the operation and control to executing fault tolerance functions.
- The components can communicate in a reliable and secure way, enabling efficient and robust information exchange and interactions that meet time-sensitive requirements.
- Fault tolerance models are designed and implemented in each component ensuring compatibility among different components, thereby facilitating data assessment through the exchange of information.
- The proposed approach for FT is designed to be generic, allowing it to be instantiated across various CPS applications. This versatility enables the proposed approach to accommodate different models of fault detection, diagnosis, and recovery, implemented in each agent.

1.4.2 Boundaries

With the purpose of delimiting the research field, clarifying the scope of the study, ensuring that the work remains focused and viable, and clearly defining the contribution it will offer to the academic and industrial community, the following boundaries were established:

- The proposed approach formalizes the stages of FT, namely detection, diagnosis, and recovery. However, this thesis primarily focuses on the detection and diagnosis stages, with comparatively less emphasis on the fault recovery stage. This boundary is explicitly reflected in the testing and validation phases, where the results are predominantly centered on detection and diagnosis. While the recovery stage is specified and formally described

in this document, its experimental validation falls outside the scope of this work, as it involves domains such as rescheduling and reconfiguration that extend beyond the thesis boundaries. By concentrating the experiments on detection and diagnosis, this thesis aims to establish a robust foundation for fault tolerance in CPSs, serving as a basis for future research. The complexities and potential innovations associated with the fault recovery stage remain valuable for future exploration, offering opportunities to advance this aspect in subsequent studies or theses.

- The formulation and specification presented in this thesis primarily focus on ensuring the accurate detection and diagnosis of faults in CPS. However, the scope does not address hard real-time requirements. Consequently, no temporal criteria were formulated within the proposed solution. Consequently, CPS applications that demand safety-critical fault tolerance fall outside the boundaries of this work. By emphasizing the accuracy and precision of fault detection and diagnosis and the impact of collaboration on these processes, this research aims to establish a robust foundation for collaborative FT in CPS. This foundation can be a stepping stone for future research and investigations into real-time and safety-critical CPS applications.

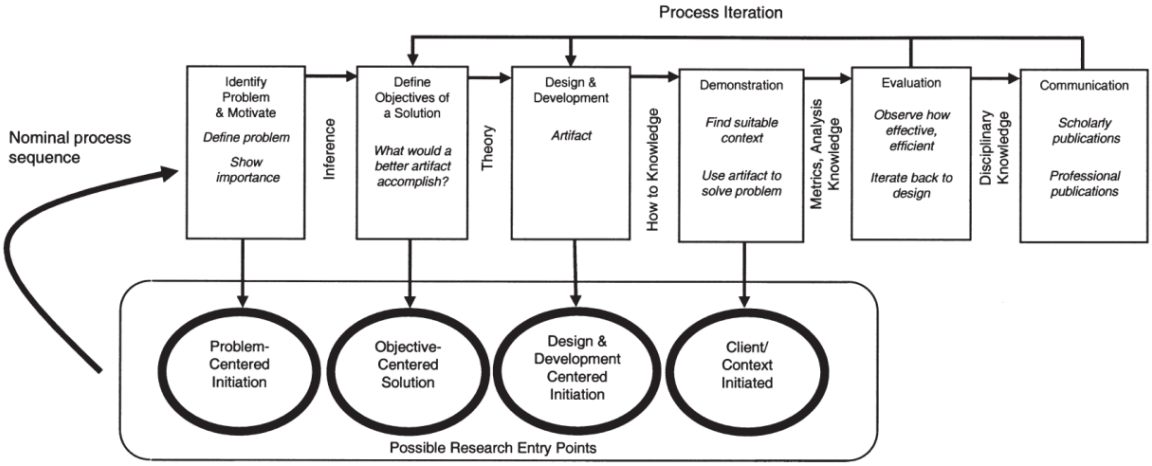
1.5 RESEARCH METHODOLOGY

This thesis adheres to the classical design science research methodology, based on approach proposed by Peffers *et al.*. This methodology, presented in Figure 1, outlines a standard sequence of six steps for developing and assessing artifacts to address a specific issue. An artifact encompasses any material created to address a problem within a particular context, including constructs, models, methods, instantiations, and processes (PEFFERS *et al.*, 2007).

This research methodology for develop the thesis follows an interactive procedure, i.e., at any phase, it is possible to evaluate the outputs, and determine if it is necessary to return to previous phase to refine or improve the artifacts (PEFFERS *et al.*, 2007). According to this methodology, the phases are applied in this research as described:

- *Phase 1 - Identify Problem & Motivate:* comprises a literature review, where, besides contextualizing the domain, it requires to define a specific research problem and provide the motivations, regarding the importance and value of its solution (PEFFERS *et al.*, 2007). In this thesis, the problem was identified through the constant analysis of related works

Figure 1 – Classical design science research methodology.



Source: Peffers *et al.* (2007).

carried out in Chapter 2, with the aim of understanding the current state of fault tolerance in the context of CPS. With this, it was possible to identify the problem, presented in Section 1.2.

- *Phase 2 - Define objectives of a Solution:* from the research problem, deduce the objectives and requirements, taking into account what is achievable and practical, to develop an artifact that can solve the identified problem (PEFFERS *et al.*, 2007). Then after completing phase 1, regarding the misalignment of FT integration in the CPS, the objective was defined, addressed in Section 1.3, proposing an approach that meets the CPS requirements for fault tolerance.

- *Phase 3 - Design & Development:* this phase is specific to transform the artifact into some practical solution (PEFFERS *et al.*, 2007). In this thesis, this phase is described in Chapter 3 with the proposed approach, the CF*, to implement FT in CPS.

- *Phase 4 - Demonstration:* involves applying the artifact in one or more scenarios or case studies to demonstrate its use in solving the research gap (PEFFERS *et al.*, 2007). In this work, the demonstration comprises a set of experiments applied in a laboratory testbed, presented in Chapter 5.

- *Phase 5 - Evaluation:* is related to assessing the outcomes of implementing the artifact and evaluating its alignment with the proposed objectives, utilizing either quantitative or qualitative metrics (PEFFERS *et al.*, 2007). In this thesis, the evaluation consist in compare

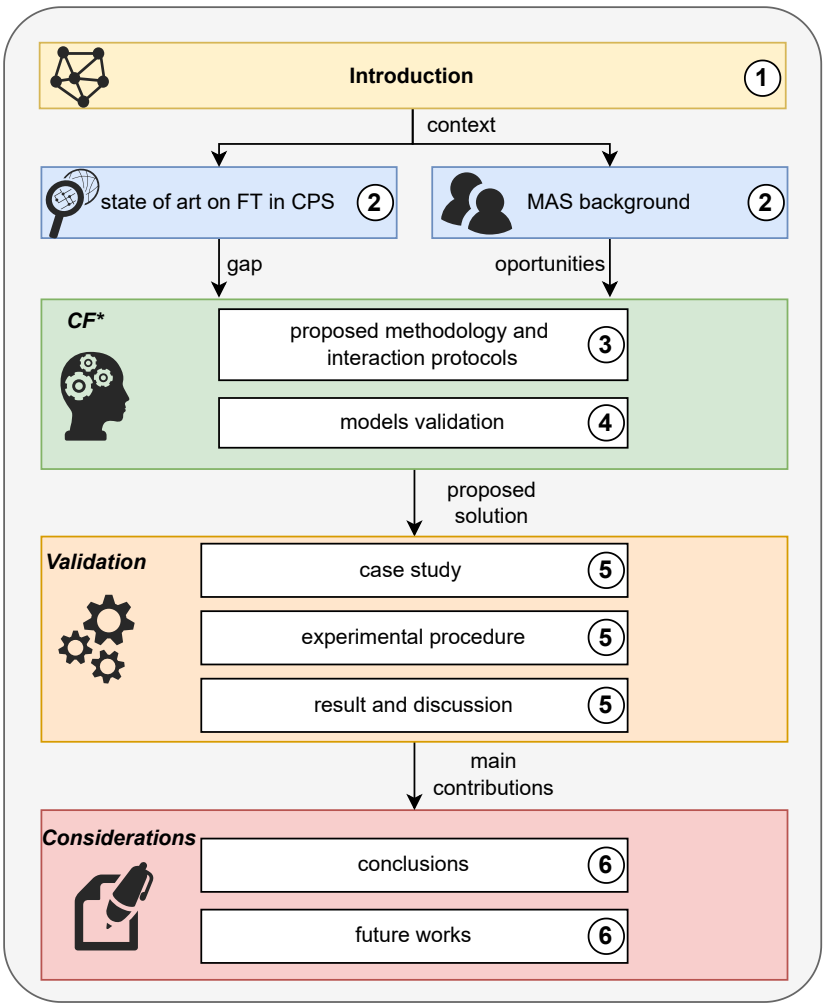
the proposed approach with traditional approach and is performed in conjunction with the previous phase (i.e., demonstration), presented in Chapter 5.

- *Phase 6 - Communication:* comprises the dissemination of the problem, artifact and the results achieved, aiming the transfer of knowledge to researchers, engineers or other relevant audiences (PEFFERS *et al.*, 2007). The dissemination is carried out by this document itself, and also through scientific articles presented at conferences and published in journals, which cover various aspects of the work detailed within this document. The list of published conference and journal articles can be checked in Appendix A.

1.6 DOCUMENT STRUCTURE

The organization of this thesis is described in this subsection. The document consists of six Chapters organized according to Figure 2.

Figure 2 – Details of organization structure of the thesis.



The Chapter 1 describes the purpose of this thesis, and presents the problem, objective, research questions and hypotheses that this thesis will address, complemented with the assumptions and limitations, and finally the research methodology adopted to develop this study.

Chapter 2, entitled “Related Work” presents the research background of this thesis. This Chapter is divided into two parts, the first part contains a comprehensive background related to FT, CPS and MAS. The second part provides a characterization of the literature connecting these three subjects, elucidating the main technologies, techniques, and domains, discussing the main approaches and methods to obtain FT in CPS, followed by an exposition of the challenges and research opportunities inherent in this area of study.

Chapter 3, entitled “Collaborative Fault Tolerance for Cyber-Physical Systems” describes the properties of collaborative FT, presenting the components that integrate the agents, as well as the internal behavior of the agents in each stage, namely, detection, diagnosis and recovery, to ensure each agent to be self-fault tolerant when possible however collaborates to improve its performance when necessary. Additionally, the Chapter presents the collaborative parameters to be implemented in each agent, the CF* meta-model, the proposed protocols and metrics.

Chapter 4, entitled “Agent Model and Dynamic Behavior Specification for CF*”, discusses the specification of the CF* approach. The dynamic behavior of each FT stage is formally modeled using the Petri net (PN) formalism for CPS purposes. Additionally, this Chapter includes the formal validation of the PN, ensuring the correctness of the models and specifications.

Chapter 5, entitled “Experiments and Results”, describes the experimental procedure and the implementation of the CF* approach in a laboratory workbench, instantiating different models of fault detection and diagnosis in the agents, as well as different variations of the collaboration parameters, comparing with centralized and local methodologies for validate the proposed methodology related to FT in CPS.

Chapter 6, entitled “Conclusions and Future Works”, concludes this document by presenting the key findings and major contributions of the thesis. It also provides a summary of proposed future developments associated with the methodology discussed, along with some recommendations for upcoming research directions in FT in CPS.

Supplementary materials developed within the scope of the thesis are presented in the appendices.

2 RELATED WORK

In recent decades, computers have evolved from expensive devices employed by governments and big corporations to widespread, mass-produced commodities embedded in human existence. Beyond their prominent presence in desktops, laptops, and smartphones, computers are embedded, often invisibly, within vehicles, household appliances, medical devices, aircraft, industrial setups, and power generation and distribution systems. Nevertheless, as the dependence on computers for the execution of essential functions increases, a substantial risk is undertaken when considering the possibility of fault in these systems, impacting the routine and activities performed by humans (KOREN; KRISHNA, 2020).

Modern systems, such as those in industry and energy generation, transmission, and distribution, demand increasingly complex techniques and approaches to ensure operational efficiency. With this complexity comes a susceptibility to faults. Despite the exponential technological advancements of the past two decades, faults, inefficiencies, and unexpected interruptions remain persistent challenges in the daily operation of systems, equipment, machines, devices, and assets. FT approaches tailored to autonomous distributed and decentralized systems, particularly from a CPS perspective, require further investigation to keep pace with the evolving technological and I4.0 requirements.

Based on that, this Chapter presents the related work of FT in CPS performed to explore the research questions of this thesis. Mainly, it discusses the background, main aspects, key technologies, and suitable approaches involved in the design and development of FT and CPS.

The remainder of this Chapter is structured as follows:

- Section 2.1 contextualizes and discusses the FT background presenting the taxonomy adopted and the fault dimensions.
- Section 2.2 introduces the concept of CPS, along with key enabling technologies, application domains, and reference architectures for implementing CPS.
- Section 2.3 presents and discusses FT in the context of CPS, characterizing the literature based on a Scopus database and analyzing the main approaches and methods adopted.
- Section 2.4 discusses MAS and discusses why this technology is a suitable infrastructure to support FT in CPS.

- Section 2.5 presents the key challenges of FT in CPS and discusses the specific gaps addressed by the research developed in this thesis.
- Finally, Section 2.6 summarizes the content covered in this related work chapter.

2.1 FAULT TOLERANCE BACKGROUND

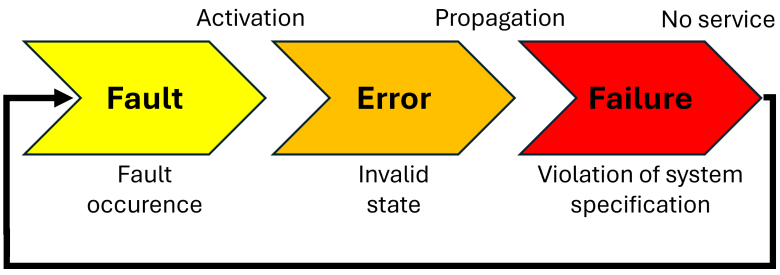
FT is a fundamental concept in computer science and engineering, particularly pertinent to the design and operation of systems that must be functional despite the faults occurrence. FT refers to a system's ability to continue operating, even in the presence of hardware, software, or human-induced faults. It encompasses a range of strategies and mechanisms designed to maintain an acceptable level of operational performance under fault conditions. The literature presents diverse FT approaches, covering a broad spectrum of applications and incorporating innovative techniques and robust methodologies, each tailored to specific contexts and system architectures.

A fault is the identified or hypothesized cause of an error (LAPRIE, 1992), sometimes referred to as a bug. It can also be seen as the consequence of a failure in another system (including the developer) that has delivered or is currently delivering a service to the given system (LAPRIE, 1995). An active fault is characterized by its ability to induce an error, as illustrated in Figure 3.

An error represents a deviation in the system's state that has the potential to cause a failure (LAPRIE, 1995). It may remain undetected (i.e., latent) or be identified (PULLUM, 2001). Errors can propagate within the system, leading to additional errors. The detection of an error implies the existence of an underlying fault.

A failure occurs when the system's output deviates from the expected behavior, commonly called an incorrect outcome (LAPRIE, 1995). The expected behavior is typically defined by a specification or a set of requirements (PULLUM, 2001).

Figure 3 – Cycle and relation between fault, error and failure.



Source: Adapted from Rehman *et al.* (2019)

Faults can cause errors and lead to single or multiple failures (AVIZIENIS *et al.*, 2004). The purpose of FT is to prevent failures by tolerating faults whose occurrences are known when errors are detected. The cycle represented in Figure 3 illustrates this general causal relationship. For example, consider a mobile robot that relies on a mathematical function to calculate the correct rotation angle based on the sine of a given value. However, due to an implementation fault, the function used in the code calculates the cosine instead of the sine. The error manifests when the function is invoked and returns an incorrect value, as the computed result does not match the expected one. Consequently, the robot follows an incorrect trajectory, representing a failure, which may lead to collisions with obstacles or failure to reach its intended destination.

Recent FT techniques and methodologies advancements have been applied across various fields and domains, including industry, transportation, aerospace, energy, distributed computing, and sensor networks.

FT in the industrial domain is crucial for ensuring the reliability and operability of systems, particularly in the context of the recent industrial revolution, i.e., the I4.0. As industries increasingly adopt advanced technologies, robust FT mechanisms become paramount to mitigate failures and maintain continuous operations. In modern manufacturing, FT remains a complex challenge, particularly in autonomous and distributed systems. Research highlights the need for innovative FT techniques that adapt to evolving manufacturing scenarios and distributed architectures (PIARDI *et al.*, 2020).

In the transportation sector, particularly in high-speed rail systems, FT is essential for safety and operational efficiency. Kong and Zhang (2021) present a distributed cooperative sliding mode fault tolerant control method for multiple high-speed trains, leveraging an actor-critic neural network to simplify the design complexity and enhance performance under fault conditions. Similarly, Zhang *et al.* (2019b) explores adaptive fault tolerant control for nonlinear flexible aircraft wing systems, emphasizing the need for stability and performance during unexpected failures. These methodologies illustrate the growing focus on adaptive control strategies that respond dynamically to fault episodes.

Aerospace applications also heavily rely on FT, especially in navigation systems. Zhang *et al.* (2019a) discuss an adaptive multi-source information fault tolerant navigation method that integrates inertial navigation systems with satellite and astronomical navigation, utilizing analytical redundancy to enhance reliability. Additionally, Weiser and Ossmann (2022) introduces fault detection and diagnosis algorithms to enable the aircraft to switch to alternate control for

high altitude laws, maintaining optimal performance during faults. These FT approaches reflect a broader trend in aerospace engineering towards integrating multiple redundant systems to ensure continuous operation despite individual component failures.

FT in the energy domain is crucial for ensuring the reliability and efficiency of energy systems, maintaining the generation, transport, and distribution functionality despite faults while minimizing energy waste (GUPTA; RANVIJAY, 2022). Various strategies have been developed to enhance FT across different energy technologies, for example, Han (2023) employ wavelet analysis technique to detect faults in voltage and frequency regulation, enabling the design of distributed fault tolerant controllers that maintain system stability.

In distributed computing, FT is crucial due to the inherent complexities and potential for component failures. Mudassar *et al.* (2020) emphasize the challenge of handling massive data generated by distributed systems, such as IoT, and discuss the need for decentralized techniques for resource leveraging and FT in edge computing. The suggested approach enhances the reliability of tasks within a distributed edge computing environment by assessing the reliability of an individual edge node at a local level and incorporating FT mechanisms, thereby elevating the overall availability of the distributed computing system (MUDASSAR *et al.*, 2020). Furthermore, Jing *et al.* (2023) mentions and justifies the importance of fault tolerant consensus among distributed computing nodes due to the vulnerability of these systems and proposes a consensus protocol for faults in distributed edge computing.

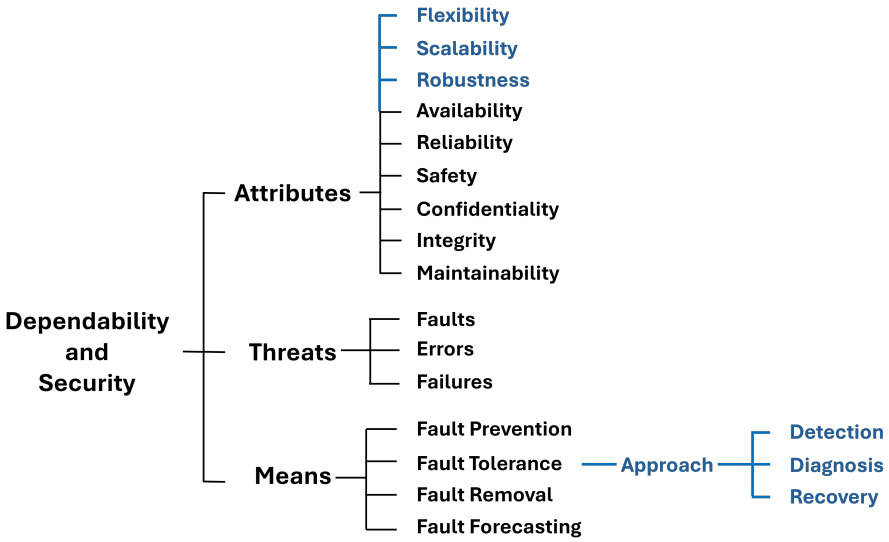
2.1.1 Taxonomy

To a comprehensive understanding of FT for this thesis, it is essential to establish a conceptual foundation encompassing classical definitions related to dependability taxonomy, the impact and modes of faults in systems, and the structural characterization of faults. In this context, the definitions of the attributes availability, reliability, safety, confidentiality, integrity, and maintainability, which are intrinsically linked to FT, are presented in this Section. These definitions, extensively recognized and adopted by the scientific community in this field, and are derived from the foundational work of (AVIZIENIS *et al.*, 2004).

Figure 4 illustrates the relationship between the concepts proposed by Avizienis *et al.* (2004), complemented (in blue text) with additional concepts relevant to modern systems and particularly pertinent to this thesis. According to Avizienis *et al.* (2004), dependability is a fundamental property of computing systems, referring to their ability to deliver reliable, safe, and

secure services despite potential faults and environmental disturbances. It encompasses multiple essential attributes for systems to function correctly and adhere to their expected behavior (AVIZIENIS *et al.*, 2004).

Figure 4 – Concept relationship tree for the thesis. Added terms are in blue.



Source: Adapted from Avizienis *et al.* (2004)

The factors that can compromise system dependability, i.e., its ability to deliver the expected behavior correctly, are faults (the root causes of system deviations), errors (the manifestations of faults in the system state), and failures (the observable deviations from correct service). These concepts have been discussed earlier in this Section and are illustrated in Figure 3.

In the context of dependability, the attributes define the essential aspects that ensure a system operates reliably, securely, and efficiently. Initially, the work of Avizienis *et al.* (2004) introduced and focused on availability, reliability, safety, confidentiality, integrity, and maintainability. However, with the advent of modern systems, an update is proposed to incorporate additional attributes that address contemporary paradigms. The emergence of new technologies, such as IoT and AI, has significantly enhanced the intelligence of systems operating in dynamic and unpredictable environments. In such contexts, autonomy, adaptability, resilience to unexpected conditions, and the ability to scale efficiently have become fundamental requirements. Three additional attributes have been introduced to accommodate these evolving demands, as illustrated in Figure 4: flexibility, robustness, and scalability.

Therefore, these attributes are defined as follows:

- *Availability*: Ensuring the system is operational and provide service when required.

- *Reliability*: Guaranteeing the ability to perform correctly and continuously over a specified period without failures.
- *Safety*: Preventing catastrophic failures, ensures that the system do not cause harm to users, the environment, or itself, even in the presence of faults.
- *Integrity*: Protecting against unintended modifications.i.e., system's state and data remain unaltered unless modified by authorized actions.
- *Maintainability*: Facilitating modifications and repairs, i.e., the system can be updated, repaired, or modified while minimizing downtime and operational disruption.
- *Confidentiality*: Ensuring data protection and privacy.
- *new - Flexibility*: Ability to adapt to changing conditions, including variations in workload, operational environments, and fault conditions.
- *new - Robustness*: Capability to withstand unforeseen faults without suffering severe performance degradation.
- *new - Scalability*: Ability to maintain the behavior and efficiency as the number of components, agents, or computational demands increase.

Additionally, FT includes several well-established traditional approaches that have been widely disseminated. Some examples include checkpoint and rollback, which periodically saves system states and restores them upon failure (EGWUTUOHA *et al.*, 2013); redundancy, where additional components are deployed to serve as backups in case of failure (GUPTA; VAIDYA, 2020); job migration, which shifts tasks to alternative computing nodes when a failure occurs (CHERAGHLOU *et al.*, 2016); and load balancing, which dynamically distributes tasks to prevent overloading specific system components (ATTALLAH *et al.*, 2021).

Beyond these approaches, fault detection, diagnosis, and recovery are critical processes that ensure a system maintains functionality despite faults. Recent advancements in methodologies and techniques for these processes have significantly enhanced modern systems' reliability, robustness, and overall performance.

Fault detection, the initial stage of FT, is responsible for identifying the presence of a fault. Its primary function is promptly signaling that something is wrong with the system, potentially leading to undesirable or unacceptable behavior (SARTORI *et al.*, 2012).

Fault diagnosis is the process of determining the characteristics and nature of detected faults and consists of accurately identifying and classifying different types of faults. For example, fault diagnosis can determine the type, location, magnitude, and cause of a fault (SARTORI *et al.*, 2012). By quickly identifying and classifying faults, the diagnosis facilitates corrective actions, mitigating the fault's impact on the system and contributing to the overall process (ISERMANN, 2006; ROJAS *et al.*, 2022). If faults are not correctly detected and diagnosed, they can rapidly escalate into more critical problems, including catastrophic failures, fault propagation between components and sub-components, and an unreliable system state (PIARDI *et al.*, 2025a).

Once a fault is detected and diagnosed, the next stage is fault recovery, which involves implementing strategies and actions to restore system functionality. In other words, this stage consists of taking appropriate actions, such as stopping operations, modifying system behavior, reconfiguring components, performing maintenance, or repairing faulty parts, either automatically or manually, to restore the system's ability to perform its intended function (SARTORI *et al.*, 2012).

These FT stages are the foundation for formulating the collaborative approach proposed in this thesis, providing the necessary conceptual basis for developing more robust and adaptive mechanisms.

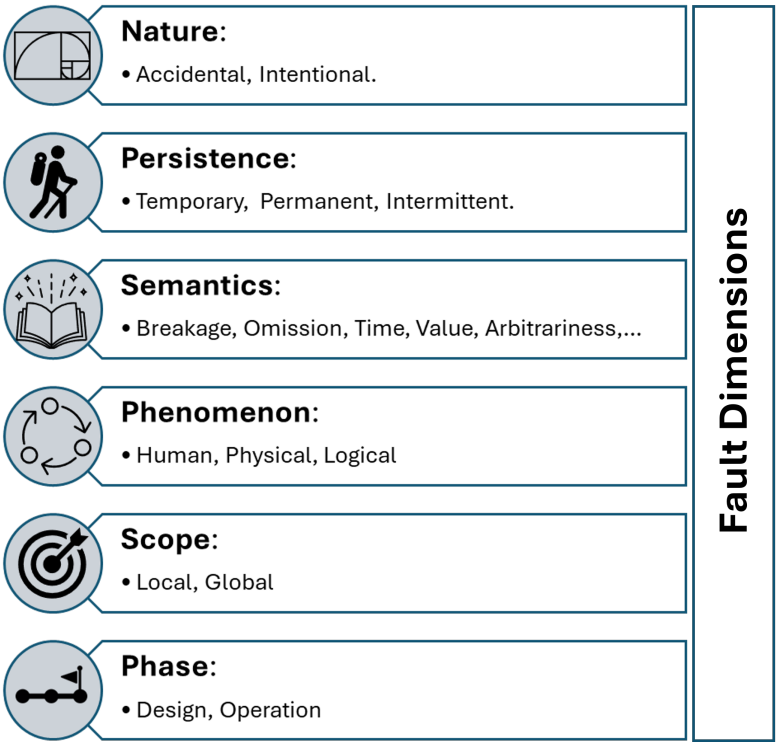
2.1.2 Fault Dimensions

System faults can be defined from various perspectives, depending on the specific domain of knowledge or application involved. Figure 5 delineates the dimensions and respective classifications constituting a fault, aiming to provide a comprehensive characterization. It is crucial to underscore that within a specific fault dimension, a fault is not necessarily confined to a single category or type, rather, it may manifest attributes that pertain to multiple categories.

The nature dimension describes whether a fault is accidental or intentional. Accidental faults often result from human error, unforeseen circumstances, or environmental factors. These faults are typically unpredictable and may arise due to a lack of knowledge, oversight, or unexpected system interactions. In contrast, intentional faults are caused by deliberate actions, such as sabotage, malicious attacks, or purposeful neglect. These faults are often premeditated and may stem from external threats or internal conflicts.

The persistence dimension represents the duration or recurrence of a fault. A fault can be classified as permanent, temporary, or intermittent. Permanent faults persist once they

Figure 5 – Fault Dimensions.



occur, requiring corrective action to restore system functionality. They may result from hardware or software design defects, physical component failures, system breakdowns, permanent link failures, or aging effects over time. Temporary faults manifest in the system for a relatively short period, often self-correcting, and could potentially repeat. These faults can be triggered by network congestion, environmental changes (e.g., weather conditions), electromagnetic interference, or transient radiation effects. Intermittent faults persist for more extended periods compared to temporary faults. They cause unpredictable behavior in the system but eventually normalize without intervention. Examples include faults caused by temperature fluctuations affecting hardware components, short circuits due to specific inputs, software reconfiguration errors, electromagnetic interference, and incorrect data computation (MELO; AQUINO, 2021).

The semantic dimension categorizes faults based on the system's behavior when a fault occurs. Understanding these classifications allows system designers to anticipate how the system will respond if such faults are triggered.

The phenomenon dimension identifies the root cause of a fault. Human faults result from errors, negligence, or intentional actions. These faults often stem from mistakes, inadequate training, or deliberate sabotage. Physical faults originate from material properties, environmental conditions, or physical processes, such as mechanical wear, corrosion, or extreme temperatures. Logical faults arise from errors in reasoning, software logic, algorithms, or design assumptions.

These faults are commonly associated with coding errors, system logic flaws, or incorrect requirements.

The scope dimension defines the impact range of a fault. A local fault affects only the specific system where it occurs, while a global fault impacts interconnected systems, potentially propagating across multiple components or subsystems.

The phase dimension identifies the stage in the system life cycle where the fault occurs. Design faults originate from errors made during the system's design phase. These faults are often due to incorrect assumptions, inadequate testing, or poor architectural planning. Operational faults emerge during the system's usage phase. They may result from usage patterns, maintenance issues, or external influences affecting the system throughout its life cycle.

By combining these dimensions, faults can be systematically classified and analyzed. For instance, a temporary, accidental, wear-related, physical, local, or operational-phase fault could describe a component that degrades over time due to normal usage but can be easily replaced. This multidimensional approach provides a robust framework for understanding, detecting, diagnosing, and mitigating faults across various domains.

2.2 FUNDAMENTALS OF CYBER-PHYSICAL SYSTEMS

With the progressive advancement of Information and Communications Technology (ICT), along with improvements in sensing, digitization, communication, computation, data analysis, and control technologies, these elements have become the foundational pillars of CPS (KUMAR; BHOWMIK, 2024). As a result, CPS has emerged as a new generation of intelligent systems that integrate physical and digital technologies, enabling more efficient and effective automation, control, and decision-making across various domains (COLOMBO *et al.*, 2017; MONOSTORI *et al.*, 2016). CPS combines sensors, processors, actuators, communication networks, and cloud computing to create systems capable of sensing, analyzing, and responding to changes in both physical environments (RAJKUMAR, 2012) and cyber computing layers (PIARDI *et al.*, 2020).

Beyond integrating physical and digital technologies, CPS leverages advanced algorithms and software to process vast amounts of digital data, enhancing autonomy and optimization in machines and processes. This integration not only fosters the development of more intelligent systems and automation but also provides engineers and users with tools to support decision-making and task execution, ultimately optimizing system behavior to achieve specific objectives

(FEI *et al.*, 2019; WANG *et al.*, 2018; WUEST *et al.*, 2016).

The evolution of CPS has opened new possibilities for automation, improved efficiency, and advanced decision-making across a wide range of industries and applications (PIARDI *et al.*, 2020). With this context in mind, this Section will present key definitions, enabling technologies, applications domains, and reference architectures of CPS.

2.2.1 Definition and Components of CPS

The foundational definition provided by National Institute of Standards and Technology (NIST) states that "CPS are smart systems that include an engineered, interacting network of physical and computational components" (GRIFFOR *et al.*, 2017). Other definitions, particularly in the industrial domain, describe CPS as systems of systems composed of a distributed network of autonomous entities, such as devices, machines, products, and services, that integrate cyber and physical components. These entities interact and collaborate to enable more intelligent, pervasive, and adaptive monitoring and control of the physical environment and its elements (HU *et al.*, 2016; KHAITAN; MCCALLEY, 2014).

CPS comprises networks of interconnected and interacting components that seamlessly integrate physical and cyber domains, combining real-world elements with software applications to create more sophisticated systems that can operate autonomously according to the context. These components, called cyber-physical components, include physical-world elements such as devices and equipments, equipped with digitization capabilities and interfaces for interacting with the environment (QUEIROZ *et al.*, 2021). Moreover, the ability of cyber components to exchange information facilitates collaboration, which is crucial for distributed and decentralized systems, enhancing scalability, flexibility, and robustness. The primary objective of cyber infrastructure is to enable an intelligent analysis of the physical environment by translating real-world data into cyber representations and to facilitate control by transmitting computational decisions back to the physical domain (HU *et al.*, 2016).

The coupling between cyber and physical elements can be observed at different scales, ranging from individual embedded systems to large-scale IoT applications, where smart physical devices are connected to cyber systems (MONOSTORI *et al.*, 2016; HU *et al.*, 2016). However, unlike embedded systems and traditional IoT architectures that merely establish connections between physical and software elements, CPS emphasizes a higher level of integration, co-ordination, cooperation, and autonomy among components (FEI *et al.*, 2019; COLOMBO *et*

al., 2017). In this context, a cyber-physical component can be a physical entity equipped with network interfaces and computational capabilities, such as an IoT smart device or sensor, or a virtual entity, such as a software system, that is tightly interconnected with physical elements, continuously interacting with and supporting both physical and cyber components.

Building upon this foundation, IoT provides essential technologies for CPS development, including sensing device platforms and standardized communication protocols that enable interoperability and compatibility between physical and cyber systems (XU *et al.*, 2018a). However, unlike conventional IoT applications, CPS conceptualizes components endowed with embedded processing capabilities, which facilitate the analysis and interpretation of their data while enabling interaction and collaboration with other components across both vertical and horizontal dimensions. These characteristics make CPS well-suited for addressing challenges in complex, large-scale systems across various domains, for example, smart factories, smart cities, smart grids, logistics, and transportation (CAO *et al.*, 2021; CHEN, 2017).

To understand the implications of CPS in industrial automation and manufacturing, it is essential to examine its relationship with I4.0. The concept of I4.0 was first introduced at the Hannover Fair in Germany in 2011 (XU *et al.*, 2018b). The German government launched the "Industrie 4.0" initiative with the primary objective of advancing digital manufacturing. According to Kagermann *et al.* (2013), I4.0 represents a shift in the manufacturing paradigm, establishing a new business model centered on product customization and digitization throughout the entire product life-cycle by integrating ICT technologies. From a broader perspective, the fundamental transformations distinguishing traditional manufacturing models from those grounded in I4.0 principles encompass improved flexibility, modularity, and reconfiguration of systems. These transformations facilitate individualized product manufacturing, comprehensive information integration throughout processes, and the seamless interconnection between physical and digital spheres.

In this context, CPS emerges as a foundational paradigm for enabling the next generation of intelligent automation and production systems. By fostering the concept of smart machines and products (LEE *et al.*, 2018; LEITÃO *et al.*, 2016), CPS is a driving force behind the digitization of traditional factories, transforming them into intelligent, flexible, robust, efficient, adaptive, and competitive production environments (KAGERMANN *et al.*, 2013; LIAO *et al.*, 2017; LU, 2017). Moreover, the integration of CPS facilitates the development of systems capable of responding in real-time and dynamically adapting to business changes and operational events, ensuring greater

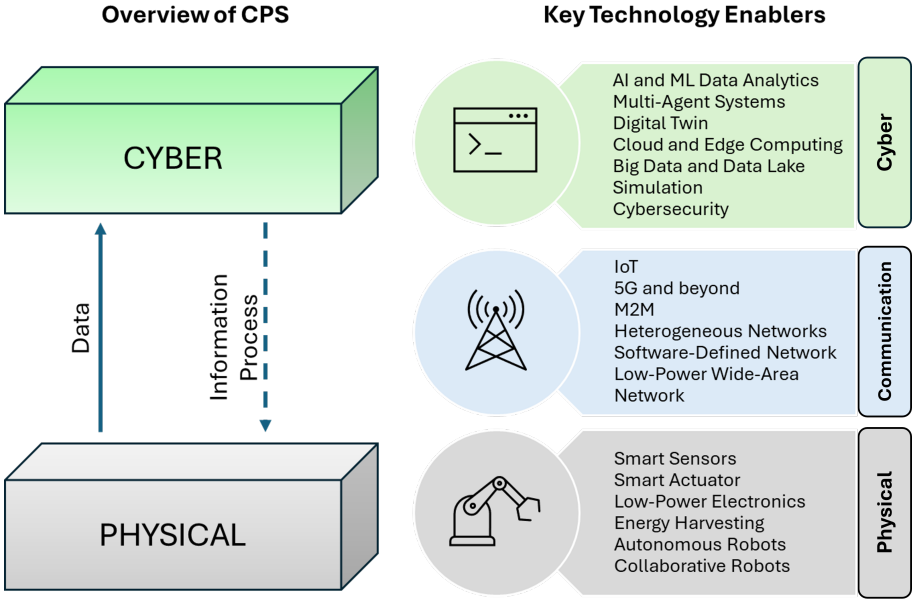
agility in industrial processes (LU, 2017; XU *et al.*, 2018b).

2.2.2 Key Enabling Technologies for Establishment of CPS

Several technological advancements are driving the establishment and evolution of CPS. For instance, sensors are becoming increasingly affordable and cheaper as they decrease in size. Additionally, breakthroughs in wireless communication, internet bandwidth, and the deployment of 5G, along with the continuous development of alternative energy sources, are further enabling the growth of CPS (KHALID *et al.*, 2021). Computing hardware is also advancing, offering higher processing capacity while consuming less power and adopting more compact form factors.

Figure 6 provides an overview of CPS and illustrates the key technologies that serve as the foundation for this paradigm. These technologies are categorized into three domains: cyber, communication, and physical.

Figure 6 – CPS with key technology enablers.



Cyber technologies form the computational backbone of CPS, enabling optimization, data storage, processing, decision-making, and intelligence through advanced computing and analytics. AI techniques, particularly those based on Machine Learning (ML) algorithms, enhance CPS by providing context awareness and advanced data analysis solutions. MAS technology facilitates the design and implementation of intelligent, autonomous, and distributed CPS applications (ABAR *et al.*, 2017). Digital Twin (DT) enables the creation of virtual representations of physical systems, improving monitoring, optimization, and system modeling (NAZARENKO;

CAMARINHA-MATOS, 2020). In the same domain, edge computing introduces computational and storage resources closer to data sources, reducing latency while improving security, energy efficiency, and overall system performance (ALI *et al.*, 2018; LI *et al.*, 2019a). Cloud computing provides large-scale data storage and analysis, and together with edge computing, it supports CPS applications in smart manufacturing and critical infrastructures (QI *et al.*, 2018). Additionally, big data and data lakes technologies allow the storage and analysis of structured and unstructured data, enriching the intelligence models implemented in the cyber layer (KHAN *et al.*, 2017). Simulation technologies are also gaining prominence, as they enable virtual replication of physical system behavior, allowing for faster and more cost-effective process optimization (LENG *et al.*, 2023). Lastly, cyber-security is a critical aspect of CPS, as the threat of cyber-attacks, along with the need for data and system protection, is particularly significant in sectors such as healthcare, energy, and transportation (ZHANG *et al.*, 2021).

Reliable communication is fundamental for data exchange between CPS components, ensuring low-latency, high-speed, Quality of Service (QoS), and secure interactions. IoT[1] functions as the core of CPS, integrating sensors, actuators, and other devices into the internet, enabling seamless data collection and transmission (LOMBARDI *et al.*, 2021; LESCH *et al.*, 2023). High-speed wireless communication, exemplified by 5G technology, is crucial in latency-sensitive CPS applications across various domains, including autonomous vehicles and smart grids (CHENG *et al.*, 2024). Machine to Machine (M2M) communication enhances interactions among CPS components, contributing to system scalability and flexibility (SINGH *et al.*, 2021). Additionally, integrating different network types, such as Software-Defined Network (SDN), Wi-Fi, and Bluetooth, is essential to maintaining seamless connectivity across diverse environments, particularly in applications spanning multiple settings (XIA *et al.*, 2022). Low-Power Wide-Area Networks (LPWAN) technology supports energy-efficient, long-range communication, making it especially suitable for CPS in scenarios requiring low power consumption over extensive distances, such as agricultural and remote monitoring applications (HERNANDEZ *et al.*, 2017).

Physical technologies are responsible for direct interaction with the environment and collecting real-world data. Smart sensor technology enhances data acquisition by enabling precise measurements, sensor fusion, and pre-processing capabilities (HA *et al.*, 2020). Aligned with energy efficiency and sustainability trends, low-power electronics play a crucial role in CPS applications that rely on remote systems (e.g., forest monitoring) and wearable devices, which

[1] Although classified within communication technologies, IoT is transversal and can integrate other technological domains

require long battery life. Complementing this, energy harvesting technologies, which capture energy from the environment, such as solar light, vibrations, or heat, offer a promising solution for powering physical devices in hard-to-reach locations where battery replacement is impractical (PENG *et al.*, 2019). While robotics has long been integrated into technological advancements, the increasing computational capabilities that enable autonomous task execution and decision-making make robots invaluable assets within CPS. Additionally, collaborative robots, or cobots, which can safely operate alongside humans, enhance human and robotic capabilities within the physical layer of CPS (GARCÍA-ESTEBAN *et al.*, 2021; NIKOLAKIS *et al.*, 2019).

2.2.3 Applications Domain of CPS

CPS, with its foundation in communication and technological integration, can encompass a wide range of application domains. Several authors have explored these domains through surveys, related works, and state-of-the-art research, providing insights into the diverse applications of CPS (GUNES *et al.*, 2014; HU *et al.*, 2016; COLABIANCHI *et al.*, 2021; BENMALEK, 2024).

Additionally, Table 1 presents a brief overview of CPS application domains, summarizing their key functionalities and mentioning recent studies pertinent to each domain.

Table 1 – CPS application domains highlighting their features and related works.

CPS Domain	Functionality	Recent Works
Manufacturing and I4.0	Optimizing productivity in the manufacture of goods or delivery of services through control, monitoring, and adaptive production systems.	(ZHANG *et al.*, 2022) (PIARDI *et al.*, 2024a)
Smart Grid and Energy Systems	Ensuring reliable and sustainable energy distribution by integrating smart meters, demand-side management, improving the coordination between grid energy sources, devices and control systems, and ensuring efficient, adaptative, and resilient energy networks	(JHA *et al.*, 2021) (NGUYEN; MOHAMMADI, 2023)
Transportation and Autonomous Vehicles	Improving mobility and safety by enabling intelligent traffic management, autonomous driving, and real-time sensor fusion for navigation and accident prevention.	(PUNDIR *et al.*, 2022) (ALSHDADI, 2021)
Healthcare and Medical Systems	Enhancing patient care and medical efficiency through remote surgery, AI-powered diagnosis, smart medical devices, and health data analytics.	(RAMUDU *et al.*, 2024) (AMINIZADEH *et al.*, 2023)
Agriculture and Precision Farming	Increasing crop yield and resource efficiency with automated irrigation, AI-driven soil and weather analytics, drone-based monitoring, and precision agriculture techniques to optimize the use of resources such as water, seeds, and pest controls.	(RIJSWIJK *et al.*, 2021) (LOPES *et al.*, 2021)
Smart Cities and Infrastructure	Enhancing urban efficiency through IoT-based monitoring, automated waste management, intelligent traffic control, and smart buildings that optimize energy consumption.	(MISHRA *et al.*, 2023) (TRICOMI *et al.*, 2024)
Aerospace and Defense	Strengthening security and operational efficiency with UAVs, aircraft, and satellites with cyber-resilient military systems and autonomous surveillance technologies.	(HUANG; LU, 2024)

The application domains of CPS are vast and diverse, showcasing its transformative impact across multiple industries and various aspects of modern life. By seamlessly integrating the cyber and physical worlds, CPS facilitates innovative solutions to complex challenges,

enhancing efficiency, sustainability, and overall quality of life. As research and technological advancements in CPS progress, its potential will further solidify its role as a fundamental pillar of the digital age.

2.2.4 Reference Architectures for Design CPS in Industrial Context

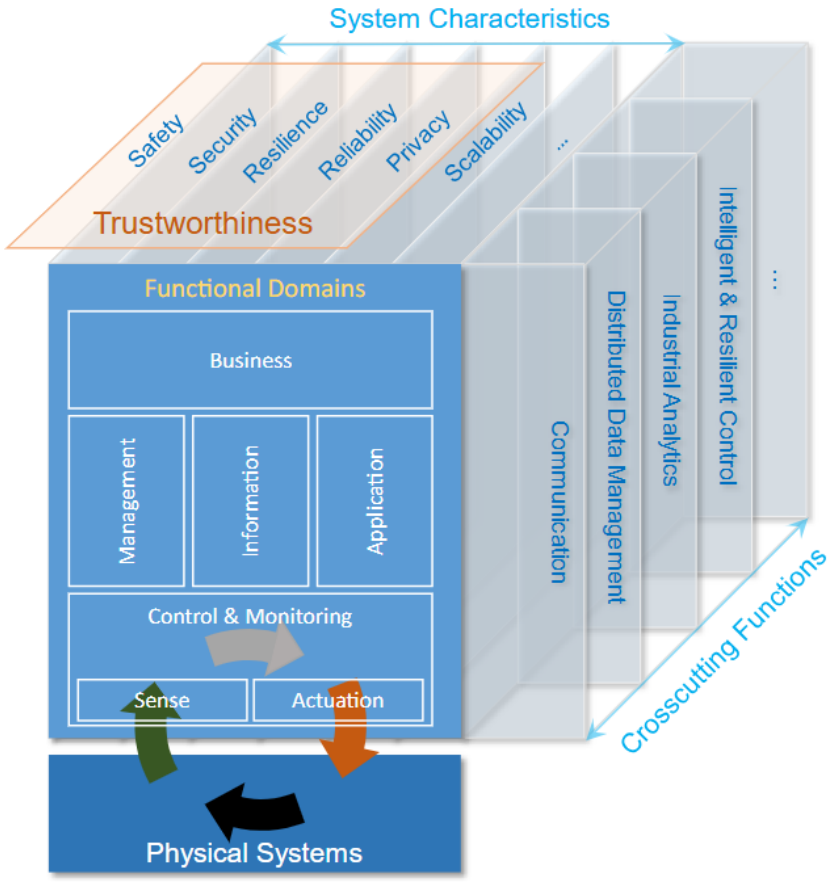
CPS has emerged as a key productivity driver across industrial sectors, especially with the advent of I4.0. Consequently, there has been a growing requirement to formulate system abstractions to standardize concepts and ensure interoperability among different systems and market players. Industrial CPS architectures serve as the framework that enables various systems and technologies to interoperate effectively. Despite the broad range of CPS application domains, the industrial sector, driven by I4.0, has strong initiatives that propose architecture models tailored explicitly for manufacturing systems based on CPS. For this reason, this Sub-section presents some of the most relevant industrial CPS architectures.

Currently, Industrial Internet Reference Architecture (IIRA) and Reference Architecture Model for Industrie 4.0 (RAMI 4.0) represent the two principal reference architectures that provide a conceptual framework for the design and development of systems and components in compliance with industrial standards, with a specific focus on the characteristics of industrial CPS and I4.0 (HABIB; CHIMSOM, 2022).

The IIRA was defined by the Industrial Internet Consortium as a USA initiative (LIN *et al.*, 2019). It is a standard-oriented open architecture that guides developing interoperable IoT systems, which can also be applied in the context of industrial CPS, following the ISO/IEC/IEEE 42010:2011 standard. This architecture provides a generic framework with a high level of abstraction and conceptual structuring for designing, developing, and implementing intelligent industrial systems, ensuring interoperability, security, scalability, and operational efficiency. As depicted in Figure 7, the IIRA structures the relationship between functional domains, crosscutting functions, and system characteristics to ensure CPS interoperability and efficiency.

The functional domains define the system's primary layers: control, operation, and analysis. The cross-cutting functions encompass security, connectivity, data management, and reliability, ensuring comprehensive system support. Meanwhile, the key system characteristics, such as scalability, resilience, and real-time performance, guarantee that the system meets industrial requirements. This integrated structure enables CPS to be modular, secure, and efficient, optimizing industrial processes. IIRA supports a wide range of application domains, including

Figure 7 – IIRA functional domains, crosscutting functions and system characteristics.



Source: (LIN *et al.*, 2019)

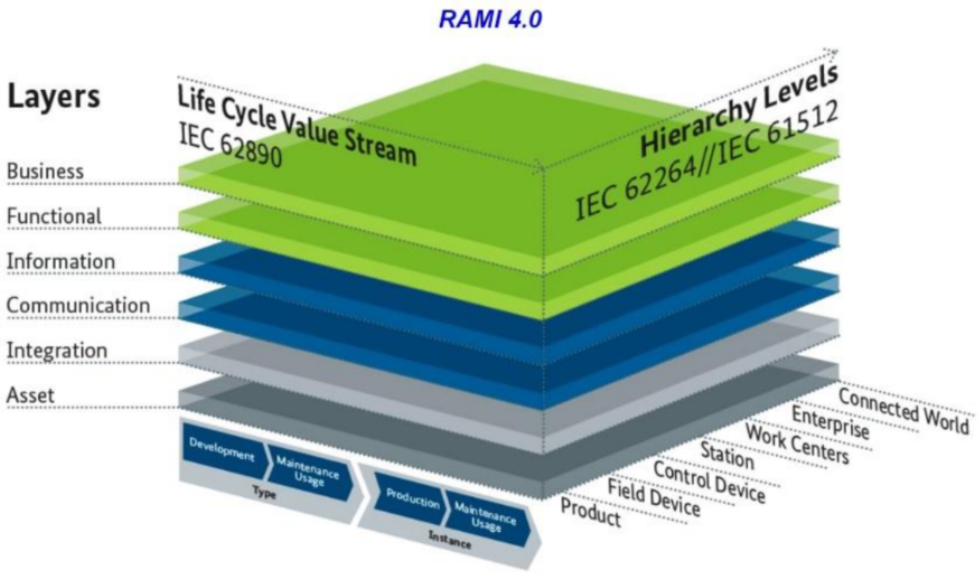
those presented in the previous Sections, and is designed for business managers, operators, planners, and other professionals seeking a deeper understanding of the benefits offered by CPS and IoT in the industrial sector (LEITÃO *et al.*, 2023).

The RAMI 4.0 is another key reference architecture, defined by Plattform Industrie 4.0 as a German initiative[2], which is still under development (ADOLPHS *et al.*, 2015). It is a reference architecture that follows a high level of abstraction and employs a standardized and unified terminology to ensure that all stakeholders involved in the design and development of I4.0 and CPS solutions maintain a consistent perspective and effectively communicate with each other. The RAMI 4.0 is represented by a three-dimensional layered model, as illustrated in Figure 8, which aims to structure I4.0 elements systematically. This model emphasizes the integration of both vertical and horizontal dimensions, thereby enhancing connectivity across the entire production workflow (LEITÃO *et al.*, 2023).

The first dimension of RAMI 4.0, labeled as layers, represents the different functional

[2] https://www.plattform-i40.de

Figure 8 – RAMI 4.0 reference architecture.



Source: (ADOLPHS *et al.*, 2015)

levels of the system, ranging from physical resources to digital abstraction. This dimension comprises six vertical layers. The asset layer represents physical entities in the real world. The integration layer facilitates the transition from the physical to the digital world through interfaces and representations. The communication layer ensures access and exchange of information between physical and digital layers. The information layer handles the representation and description of data and services. The functional layer defines the logical functions and services associated with the asset. Finally, the business layer encompasses organizational structures and business processes. This layered structure ensures a systematic approach to integrating cyber and physical elements within CPS, enabling interoperability and scalability in industrial applications (PIVOTO *et al.*, 2021).

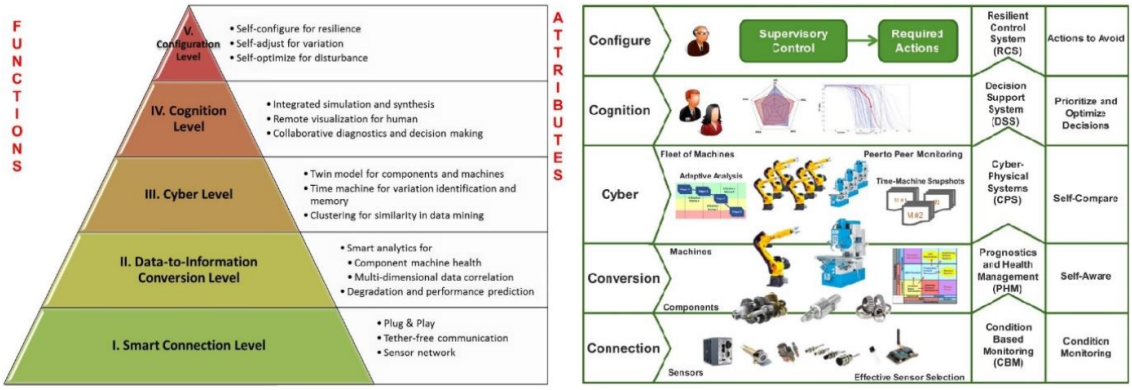
The second dimension, named the hierarchy levels, defines the product life cycle structure and the industrial automation hierarchy, covering devices, stations, cells, factories, and enterprise networks. This dimension ensures the integration between different manufacturing levels and processes in a mesh-like manner. Within this comprehensively networked setting, every component can communicate and interact across various hierarchical levels, facilitating increased adaptability and intelligence within systems while ensuring interoperability between the physical and digital worlds (PIVOTO *et al.*, 2021).

Finally, the life-cycle and value stream dimension of the RAMI 4.0 framework integrates data acquisition, integration, and seamless accessibility throughout the life-cycle of assets such as

machines and products. This integration extends the entire value chain and encompasses diverse processes and stakeholders. By leveraging the value of data, this dimension seeks to advance product-as-a-service paradigms, improving customer experiences while optimizing production and maintenance operations (PIVOTO *et al.*, 2021).

Beyond these reference architectures, the research and academic community has proposed numerous system architectures adapted to specific aspects and application fields within CPS. Notably, the 5C architecture is a prominent and widely adopted CPS architectures model, offering conceptual guidance for engineers and system developers. It is mainly dedicated to the design and development of CPS in manufacturing contexts (LEE *et al.*, 2015). Figure 9 illustrates the 5C architecture, which defines a five-level structure that addresses system attributes and functions. This architecture spans from data collection and processing levels to advanced levels of data analysis and decision-making support operations (LEE *et al.*, 2015).

Figure 9 – The 5C CPS architecture with related applications and techniques.



Source: (LEE *et al.*, 2015)

In the context of the current digital transformation, driven by CPS and supported by its key enabling technologies, the adoption of the presented reference architectures, namely IIRA, RAMI 4.0, and 5C, across different application domains, as previously referenced within this Section, can significantly contribute to the digitization of systems and assets. These architectures provide structured guidelines and best practices to meet application requirements while facilitating the development of interoperable, scalable, and efficient industrial CPS.

2.3 FAULT TOLERANCE IN CYBER PHYSICAL SYSTEMS

The high level of autonomy and the distributed nature of CPS, combined with the large number of interconnected devices, make these systems susceptible to faults. Ensuring

robustness and reliability in CPS requires that their components incorporate FT mechanisms. Fault occurrences can lead to efficiency degradation and may compromise the integrity of cyber-physical components (XU *et al.*, 2020).

Due to their distributed characteristics, CPS are inherently complex, relying on continuous information exchange to perform operations. Integrating FT into these systems is particularly challenging, as it demands computational resources and design considerations to ensure effective fault detection, diagnosis, and recovery (PIARDI *et al.*, 2020).

This Section characterizes FT in CPS, discusses the main approaches and methods and categorizes relevant studies based on the CPS domain, adopted methods, and techniques.

2.3.1 Literature Characterization

Investigating how FT enhances the robustness and resilience of CPS is essential to assess the maturity of this research field and identify gaps in the literature. This cross-domain survey examines the evolution of research on FT in CPS, identifying key attributes, techniques, enabling technologies, and the most prominent application domains.

The literature review was conducted using the Scopus database, covering contributions indexed from January 1, 2007, to December 31, 2024. Scopus was selected due to its academic relevance, indexing over 30,000 journals from over 7,000 publishers and containing more than 91 million records (WALKER, 2024). The first step in this process involved defining the scope and parameters of the search query, which aimed to identify all studies that explicitly associate CPS with FT, ensuring a comprehensive dataset covering different dimensions of FT in CPS.

The search query was formulated to include variations of both terms, as follows:

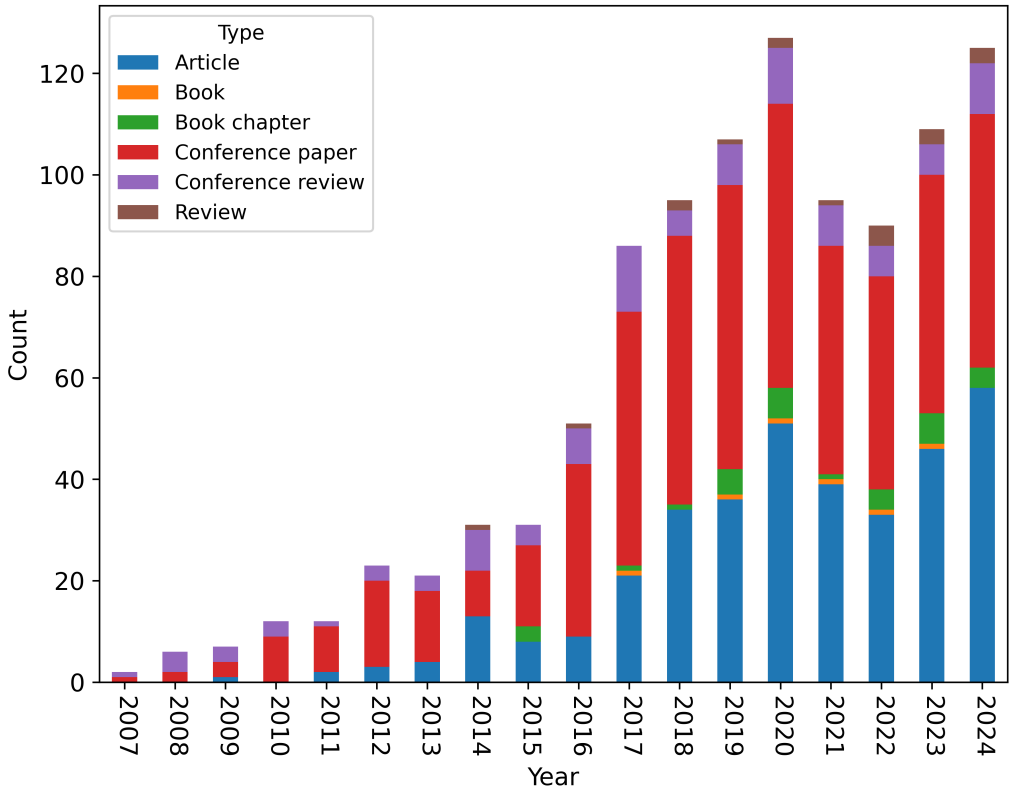
> *((“fault toleran*” **OR** “fault-toleran*”) **AND** (“cyber-physical system*” **OR** “CPS” **OR** “cyber physical system*” **OR** “industrial cyber-physical system*” **OR** “industrial cyber physical system*” **OR** “ICPS”)).*

This query captures both singular and plural terms, along with derived forms, in the titles, abstracts, and keywords of records indexed in Scopus.

After applying exclusion criteria to removing notes, editorial documents, retracted articles, and non-English documents, the final dataset comprised 1,030 papers, including journal articles, books, book chapters, conference papers, and review articles. Figure 10 illustrates the evolution of scientific publications on FT in CPS. A steady increase in research output

was observed from 2007 to 2020, followed by a decline between 2021 and 2023, potentially influenced by the COVID-19 pandemic. However, in 2024, publication levels returned to those of 2020, with journal articles achieving maximum numbers. This trend indicates the scientific community's sustained and growing interest in the relationship between these topics.

Figure 10 – Evolution of published scientific articles focusing FT in CPS over the time.



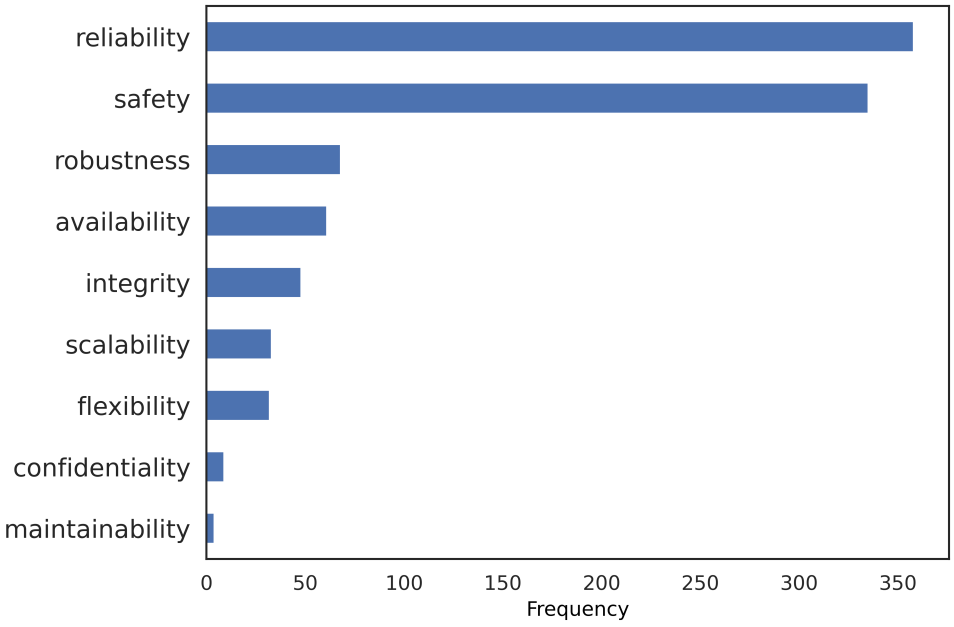
Furthermore, the analysis of n-grams, i.e., the extraction of contiguous sequences of n items (e.g., words or phrases) from texts, is performed using the Natural Language ToolKit (NLTK) library[3] in Python to map the landscape of FT research in CPS. This methodological approach serves to identify key attributes of FT applied to CPS, as well as the primary techniques, mechanisms, and enabling technologies associated with FT. Additionally, it helps to determine the most relevant application domains within CPS that exhibit significant research interest in this area. By systematically analyzing the frequency and co-occurrence of n-grams in the abstracts of papers retrieved from the Scopus database, this approach reveals patterns and trends, offering a comprehensive perspective on the predominant research and advancements in FT in CPS.

Figure 11 presents the most frequently mentioned attributes of FT based on the analyzed abstracts. Among these attributes, reliability emerges as the most prominent, followed by safety, robustness, and availability, emphasizing their role in ensuring the operation of CPS. Other

[3] NLTK documentation: https://www.nltk.org/

attributes, such as integrity, scalability, flexibility, confidentiality, and maintainability, appear less frequently but still indicate relevance in designing and implementing FT mechanisms for CPS. The prominence of these attributes highlights the multifaceted nature of FT in CPS, essential for addressing the challenges of digital environments. Moreover, the distribution of these attributes underscores the research community's focus on developing CPS capable of operating under diverse and potentially adverse conditions with FT attributes.

Figure 11 – FT attributes in CPS based on analysis of retrieved abstracts ranked by occurrence frequency.



Continuing the analysis with a focus on the means and techniques used to achieve FT in CPS, Figure 12 presents the most frequently mentioned terms based on abstracts retrieved from the Scopus database. Techniques such as detection, diagnosis, and monitoring dominate the frequency distribution, reflecting their fundamental role as the first step in identifying and addressing faults within CPS. Additionally, methods like isolation, prediction, and recovery are prominently featured, highlighting the importance of not only detecting faults but also mitigating their impact and restoring system functionality. Other techniques, including mitigation, self-healing, prognostics, correction, forecasting, prevention, checkpointing, and masking, appear with varying frequencies, indicating their relevance in specific contexts or applications. The prominence of these techniques underscores the research community's focus on developing comprehensive FT strategies that encompass fault identification, mitigation, and system recovery. Furthermore, the distribution suggests a growing interest in proactive and adaptive approaches, such as prediction and self-healing, which are crucial for enhancing the resilience and dependability of CPS in dynamic and uncertain environments, attributing autonomy and intelligence

to devices. Overall, the findings highlight the diverse and evolving nature of FT techniques, reflecting the complexity and multifaceted aspects involved in ensuring the robustness of CPS.

Figure 12 – FT means and techniques based on analysis of retrieved abstracts ranked by occurrence frequency.

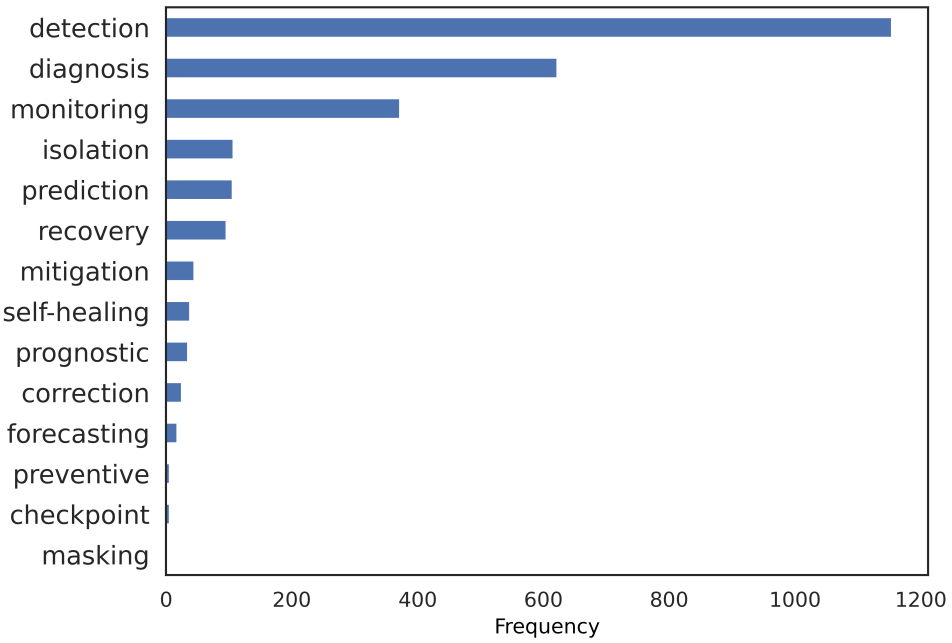
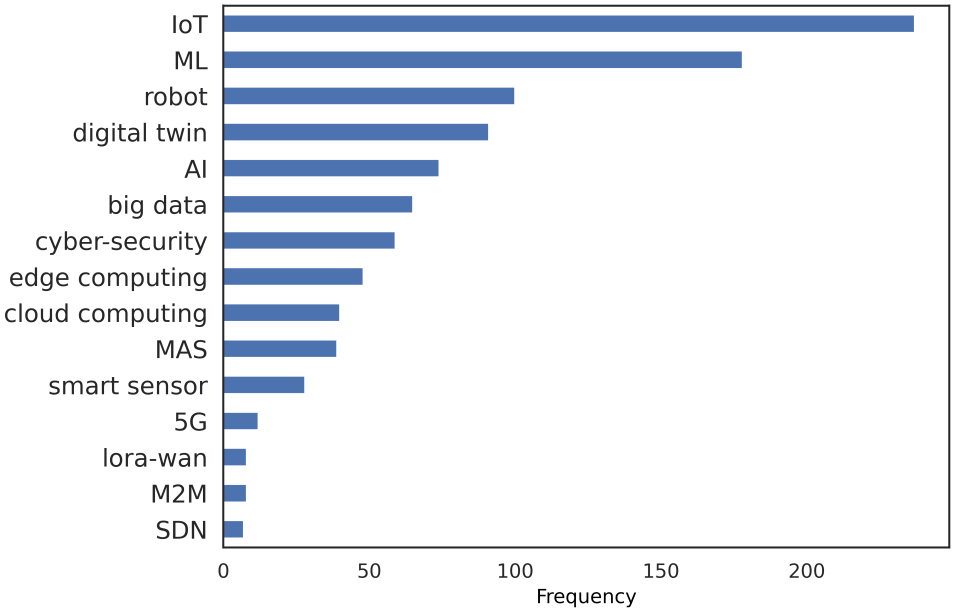


Figure 13 presents the frequently mentioned enabling technologies based on analyzing paper abstracts using NLTK. Among these, IoT emerges as the most prominent technology, reflecting its essential role in digitization by connecting physical and cyber components, consequently enabling fault analysis. Technologies such as ML, AI, and big data also feature prominently, underscoring their importance in enabling advanced data processing, decision-making, and predictive capabilities, which are crucial for FT. Additionally, digital twin, cyber-security, edge computing, and cloud computing are frequently cited, highlighting their contributions to system reliability, scalability, and security. MAS also appears prominently in the analysis, emphasizing its role in enabling distributed decision-making, intelligence, and autonomy in CPS components. By leveraging autonomous agents to manage faults, MAS enhances CPS adaptability, scalability, and FT. Other technologies, including smart sensors, 5G, LoRaWAN, M2M, and SDN, appear with varying frequencies, reflecting their specific contributions to communication, coordination, and adaptability within CPS.

The prevalence of these technologies highlights the growing scientific interest in leveraging cutting-edge advancements to improve the dependability of CPS, ensuring robust, efficient, and secure system operations. This distribution also underscores the interdisciplinary nature of CPS, where developments in communication, computation, and algorithms converge to create re-

silient and adaptive systems capable of operating in dynamic environments. Overall, the findings reinforce the role of these technologies in driving innovation and enabling the deployment of CPS with enhanced FT capabilities.

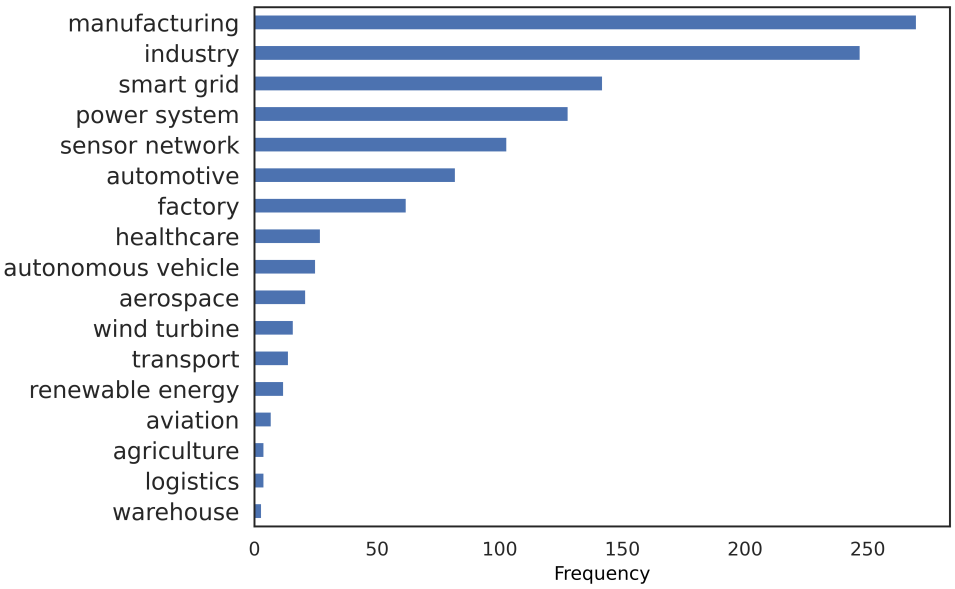
Figure 13 – Related technologies based on analysis of retrieved abstracts.



Finally, Figure 14 highlights the key CPS domains that are at the forefront of adopting FT approaches. The manufacturing and industry are prominent domains, reflecting its pioneering role in integrating FT mechanisms to ensure reliability and robustness in smart factories and automated production systems. Other cutting-edge domains, such as smart grids, power systems, and autonomous vehicles, also feature prominently, demonstrating the adoption of FT to maintain operational stability and safety in highly dynamic and complex environments. Additionally, domains such as healthcare, aerospace, renewable energy, and transportation increasingly incorporate FT capabilities into CPS, driven by the need for dependability and resilience in critical applications. The presence of these domains underscores the ongoing efforts to integrate FT into CPS, particularly in sectors where system faults can have significant economic, safety, or environmental consequences. These CPS domains are at the forefront of deploying FT strategies, leading the way in enhancing reliability, security, and efficiency.

The characterization of the literature, through the analysis of FT in CPS, has provided a comprehensive understanding of the primary domains, techniques, and technologies driving this field. This analysis serves as a foundation for the research presented in this thesis, which aims to propose a generic approach that seamlessly integrates with these technologies and is applicable across different CPS domains. Furthermore, it addresses a critical gap identified in the literature:

Figure 14 – Top CPS domains using FT based on analysis of retrieved abstracts ranked by occurrence frequency.



the need for enhanced collaboration among CPS components to improve FT capabilities. By focusing on this aspect, the proposed approach seeks to advance the state of the art in CPS by fostering more robust, flexible, and scalable systems capable of mitigating faults through interaction and collaboration among system nodes.

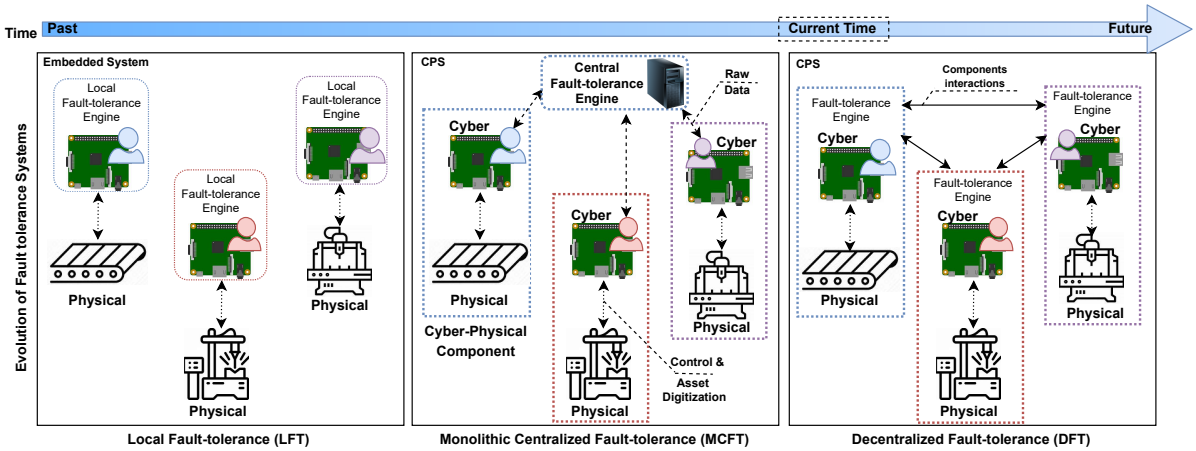
2.3.2 Related Approaches

In CPS the main FT approaches are typically deployed locally, i.e., each component locally integrates FT techniques, or is monolithic centralized, i.e., a central node is responsible for the FT of the CPS (PIARDI *et al.*, 2025b). Such practices are out of sync with the current technological reality and even with the paradigm addressed by the current CPS, which is based on distribution, autonomy, and collaboration (LEITÃO *et al.*, 2023). Although CPS adopt technologies that emerged with the fourth industrial revolution, implementing FT approaches locally or centrally introduces a single fault point and is consequently limited in achieving the maximum benefits enabled by the integration of these technologies, such as improvements in accuracy, performance, scalability, and flexibility.

As highlighted in several surveys (BAYAR *et al.*, 2015), there is a lack of formalized architecture that integrates and adopts collaboration between cyber-physical components as a key factor in improving fault detection, diagnosis, and recovery at the local (i.e., in each component) and global (i.e., system-wide) levels.

To structure the discussion on FT in CPS, the approaches were classified into three distinct categories: Local Fault Tolerance (LFT), Monolithic Centralized Fault Tolerance (MCFT), and Decentralized Fault Tolerance (DFT). This categorization, illustrated in Figure 15, reflects key architectural and operational differences commonly found in CPS implementations, aiming to provide a clearer understanding of existing solutions. This classification was derived from the literature characterization realized earlier in this Section and the reviews realized in previous research (PIARDI *et al.*, 2020; PIARDI *et al.*, 2023).

Figure 15 – Timeline of different approaches for fault tolerance in the context of cyber-physical systems.



LFT is a FT approach for embedded systems that integrates independent detection and diagnosis techniques for each device (SAFARI *et al.*, 2022). Therefore, each component is locally responsible for detecting and diagnosing faults based on integrated techniques that analyze the individual device context and generate alerts for the maintenance team to conduct the recovery stage. There is no information exchange or interaction to mitigate faults. Several applications adopt this approach due to its simple development, for example, the development of machine-condition monitoring embedded in low-cost micro-controllers (GRETHLER *et al.*, 2021). Currently, LFT can integrate recent technologies, such as AI and ML, for accuracy improvement. This approach is responsive quickly because it deals with small data volumes and is suitable for scalability.

One key advantage in LFT approach is the enhanced responsiveness to faults. The LFT enable immediate detection and reactions to faults within individual components. Cassano *et al.* discuss this, highlighting that it allows for timely decision-making by system nodes, where local information about state and faults is readily available and actionable (CASSANO *et al.*, 2021). Furthermore, LFT contribute to reduced communication overhead. Relying on local components to monitor and manage faults diminishes the need for transmitting extensive data to a central or

other components. This reduction in communication conserves bandwidth and minimizes latency, as local components can act on their information without waiting for external instructions. This is especially important in systems with many interrelated parts, where frequent communication in a centralized or distributed system can lead to bottlenecks (BADR *et al.*, 2024). Additionally, the independent nature of LFT minimizes the risk of single points of fault (BEG *et al.*, 2021). This approach allows the system to maintain operations even when an individual component fails. Zhao *et al.* demonstrates how localized control mechanisms enable components to remain functional and contribute to overall system performance despite faults (ZHAO *et al.*, 2023). However, the LFT has drawbacks, such as when a global or generalized fault occurs, that is, a fault with the same nature is detected, diagnosed, and reported by several system components, or restricted awareness presenting difficulties in adapting to situations in which operating conditions change due resulting in inconsistent results due to myopia and lack of context.

With the advent of CPS, centralized industrial systems have been enhanced, and the MCFT approach has gained prominence and become popular for FT in CPS (MOUSSA *et al.*, 2019; REIJNEN *et al.*, 2021). Each physical component has its data digitized to the cyber part of the system through sensors that adopt IoT technology. The cyber layer processes and analyzes these data to control their physical components, equipped with high computational power, advanced techniques, and AI algorithms. In addition, the cyber layer sends raw data to a central node or server, which stores data from all components and uses different techniques to detect and diagnose faults globally. When a fault is diagnosed, the recovery stage starts; for example, executing self-calibration of control parameters or sending a report to the maintenance team to make the necessary corrections, adjustments, repairs, or replace damaged equipment (RIBEIRO; BARATA, 2011). This approach has the advantage of simple deployment and provides support easily because all stages run in a unique logical unit or server, simplifying algorithm updates and bug fixes. However, its main drawbacks include the following: (i) the central node must deal with a large amount of raw data, which can cause data loss and reduced responsiveness when detecting and diagnosing faults; (ii) difficulties in system scalability due to processing and memory limitations, which directly affect the reactivity of detection and diagnosis, as well as fault recovery; (iii) monolithic policies can compromise the accuracy and confidence of fault detection and diagnosis because of the adoption of uniform classification rules across diverse equipment, disregarding the operational context.

One notable advantage of MCFT is the consolidation of control and management

functions. A central component can synchronize the actions of multiple components within the system, to operate cohesively even in the presence of faults. By managing information and operations from a single component, the system can implement coordinated responses to FT (ZHOU *et al.*, 2012; LI *et al.*, 2019b) Furthermore, the MCFT benefit from improved data aggregation and analysis capabilities. With all relevant data channeled through a central node, it becomes easier to perform system monitoring to detect, diagnosis and recovery from anomalies.

This centralized fault management leads to more resilient systems, as detailed by Zhao *et al.*, who highlight how centralized processing can leverage extensive datasets for more accurate diagnostics. In cases of faults, the central node can utilize machine learning or statistical methods to analyze trends and predict potential failures before they occur, thereby proactively mitigating risks (ZHAO *et al.*, 2023). In addition, the MCFT facilitates easier maintenance and updates of the CPS. Upgrading or maintaining a centralized approaches can be less complex than dealing with the numerous components in a distributed solutions (GENSH *et al.*, 2017). Consequently, system operators can implement improvements or modifications more effectively, thus enhancing fault tolerance over time (CHAKRAVARTY; MAHANTA, 2016).

DFT integrates the collaboration between distributed components with FT models. This proposal implements FT methods locally for each cyber component and sends information regarding the status and context to the other components. Component resilience can be improved based on the local data received from sensors in the physical part and contextualized with data from other components. For example, multiple mobile robots share their status and context to avoid congestion in navigation during the transport of products in logistics warehouses and to avoid planning routes with waypoints where robots have stopped (i.e., failed), interrupting the path (PIARDI *et al.*, 2022). Another example is consensus-based voting to detect faults, in which edge devices with ML algorithms perform consensus voting on each data observation to detect faults in temperature sensors, thereby improving the detection accuracy compared to local approaches (FIDELIS *et al.*, 2022).

One significant disadvantage of DFT is the increased communication overhead necessary for coordinating FT mechanisms across multiple components. As discussed by Garraghan *et al.*, this approach requires complex interaction protocols that can lead to a higher resource utilization due to the substantial data exchanges between nodes (GARRAGHAN *et al.*, 2016). This overhead can be especially pronounced in larger systems, where communication is required to maintain coherent operation, ultimately contributing to latency and potential component performance

degradation (GARRAGHAN *et al.*, 2016). Another challenge is the complexity of recovery processes required in DFT. In this approach, while each component can operate autonomously, ensuring that all components remain aware of the system's overall state and can adequately respond to faults complicates the recovery process.

Ubaidillaha *et al.* emphasize that a dependable fault tolerance approach in distributed systems necessitates extensive mechanisms for fault detection, identification, and recovery, thus escalating overall system complexity (UBAIDILLAHA *et al.*, 2023). This added complexity can impede rapid responses to faults and extend recovery times. Moreover, DFT approaches may face inconsistency in how different components address the same faults, potentially resulting in suboptimal recovery strategies. When components employ varying implementations of FT, their reactions to identical fault scenarios can diverge, leading to unpredictable behavior. As Tertytchny and Michael note, the heterogeneity and scale of components in distributed CPS present challenges in defining consistent faulty behavior and responses across diverse system parts (TERTYTCHNY; MICHAEL, 2020). This inconsistency can exacerbate recovery issues and lead to inefficiencies. Finally, DFT often necessitate more rigorous deployment validation procedures than LFT or MCFT due to their inherent complexity. Coordinating multiple autonomous components under fault conditions requires comprehensive validation to ensure reliable functioning. Kalaskar and Somasundaram point out that this thorough validation is essential for ensuring that fault diagnosis and recovery mechanisms operate as intended, though it also extends overall development and deployment times (KALASKAR; SOMASUNDARAM, 2024).

While recent technological advances have paved the way for decentralized CPS, the DFT is in an embryonic phase and evolving. This requires a detailed and comprehensive specification of how a CPS can take advantage of the benefits of recent technological advances reflected in the collaborative behavior of the components. Although the potential of DFT approach, a comprehensive specification is still required to delineate collaborative behaviors of cyber-physical components thoroughly.

Despite their importance and benefits, traditional FT methodologies, namely local and/or centralized, often overlook the potential benefits of collaboration between cyber-physical components.

2.3.3 Principal Methods

Currently, the academic community is actively engaged in developing new methods, techniques, and algorithms for FT deployment in CPS applications. These efforts focus on three principal approaches to fault detection and diagnosis, namely, model-based, data-driven, and knowledge-based methods, each applied across different CPS domains (DOWDESWELL *et al.*, 2020). Table 2 categorizes these methods and associates them with their respective features and techniques.

Table 2 – Categories of FT methods for CPS.

Methods Category	**Features**	**Examples of Related Techniques**
Model-based	Relies on precise mathematical or physical models to detect and diagnosis faulty conditions, requiring accurate models and parameters, and is effective for well-understood systems with deterministic behavior.	State Estimation Kalman Filters Markov Models Fault Trees Residual Evaluation Resudial Generation Constraint Suspension
Data-driven	Focuses on historical and current stream data for FT in complex systems, adopts AI and statistical techniques, and adapts to changing system conditions without explicit models. Requires datasets for training and validation.	Artificial Neural Networks Machine Learning Fuzzy Logic K-Nearest Neighbors Support Vector Machines Principal Component Analysis
Knowledge-based	Utilizes expert knowledge, rules, and heuristics for FT, incorporating domain-specific insights and human expertise. It is effective for systems with fault patterns. Does not require extensive data or precise models.	Bayesian Decision Rule Based Reasoning Case Based Reasoning

The model-based method mathematically or computationally models the cyber-physical components to represent their expected behavior. The parameters computed by the model are then compared with real system observations (SKLIROS *et al.*, 2018). When discrepancies arise between the model and actual behavior, fault detection, diagnosis, and correction mechanisms are triggered. Techniques such as Markov models, Bayesian networks, and Kalman filters are commonly employed in this approach to enhance FT in CPS.

FT conducted by data-driven methods relies on analyzing data collected from cyber-physical components or the CPS as a whole. These techniques establish correlations between observed measurements and AI or ML models. In the case of statistical approaches, they assign a probability to the system's current state (JIANG *et al.*, 2018). Machine learning techniques such as neural networks, Support Vector Machine (SVM), or K-Nearest Neighbors (KNN) are commonly employed to detect anomalies and diagnose faults before they lead to significant disruptions (CHEN *et al.*, 2023). This approach is particularly valuable in complex CPS, where

obtaining precise mathematical or computational models is challenging or when faults do not exhibit a uniform pattern.

Finally, the knowledge-based method employs prior knowledge and heuristic rules, relying on fact-based or inference-driven reasoning to manage faults (CHI *et al.*, 2022). Approaches such as rule-based reasoning and case-based reasoning are applied to identify and address faults based on past experiences and predefined rules. This method is particularly relevant in domains where human expertise is crucial for decision-making related to malfunction in the CPS.

To provide an overview of the research landscape on FT in CPS, a review of relevant studies in the field was conducted. Table 3 presents a sample of key articles exploring different FT methods in CPS, highlighting their main characteristics. This analysis serves as a foundation for understanding how different CPS domains adopt various FT approaches, methods, and techniques, as well as their contributions to improving the resilience and robustness of CPS.

Table 3 – Summary of studies with FT approaches, methods, and techniques in the context of CPS.

Studies	CPS Domain	FT in CPS Approach	FT Method	FT Technique	Brief Study Description
(SHAO *et al.*, 2019)	Industrial induction motor	LFT	Data-driven	Deep convolutional neural network	A fault diagnosis approach for induction motors using deep convolutional neural networks is presented. It leverages multiple signal inputs to enhance fault detection accuracy.
(GRETHLER *et al.*, 2021)	Industrial assets	LFT	Data-driven	Machine learning	Explores the use of embedded ML to monitor machine conditions. It emphasizes low-power, edge-based implementations for efficient fault detection and predictive maintenance in industrial settings.
(VAN; CEGLAREK, 2021)	Manipulator Robot	LFT	Model-based	Sliding mode observer	Proposes a FT control strategy for robot manipulators, ensuring stability and performance despite faults. The approach guarantees global fixed-time convergence, enhancing reliability in robotic applications
(ZHAO *et al.*, 2019)	Mobile Robot	LFT	Model-based	Extended Kalman filter	Relative and absolute localization techniques fused with Extended Kalman Filter for detecting and diagnosis faults in multi-robot localization
(BORAL *et al.*, 2019)	Machinery	LFT	knowledgement based	Case-based reasoning	A framework to detect and diagnose faults in machinery and gearbox utilizing the available historical fault data, improving diagnosis accuracy and maintenance efficiency.
(WANG *et al.*, 2014)	Smart grid	MCFT	Model-based	State estimation and residual evaluation	Proposes a sensor-based relay monitoring mechanism to enhance fault tolerance in overhead transmission lines. It improves fault detection and system reliability in smart grid applications.
(XING *et al.*, 2017)	Storage data center in IoT app.	MCFT	Model-based	Fault Tree	Propose a model for a mesh network using a fault tree to understand the logical and dynamic relationship of different IoT component failures to find an action to recover the system and enhance its resilience.
(HASEEB *et al.*, 2022)	Smart city	MCFT	Data-driven	Machine learning	Propose a model to offer fault tolerant routing for IoT devices and sensor communication in the context of smart cities. It enhances data security, reliability, and network performance in urban environments.
(MOUSSA *et al.*, 2019)	Wireless sensor networks	MCFT	knowledgement based	Case-based reasoning	Propose a centralized FT mechanism that deals with permanent and transient faults in wireless sensor networks. It improves network reliability and resilience by efficiently detecting and managing faults in sensor nodes.
(FIDELIS *et al.*, 2022)	Edge network devices	DFT	Data-driven	Machine learning	Propose a consensus algorithm to decide the result of inference in a decentralized architecture in edge computing for all inferences, where all devices have different ML models.
(XU *et al.*, 2020)	Physical actuator	DFT	Model-based	Adaptative threshold and sliding mode	Detect and diagnose faults of bias and loss of efficiency in actuators considering a network of physical devices using a distributed fault observer node.
(REPPA *et al.*, 2014)	Generic CPS	DFT	Model-based	-	Proposed distributed sensor fault detection and isolation technique for interconnected CPS in two levels, namely the local fault analysis and global fault analysis. It improves system reliability and resilience through distribution.
(LIANG *et al.*, 2024)	Cyber security	DFT	Model-based	Kalman Filter	Presents a distributed framework for detecting cyber attacks and diagnosing physical faults in large-scale interconnected systems. It enhances security and reliability through real-time monitoring and fault tolerant strategies.

These studies summarize various research efforts addressing FT in CPS, adapting methods and techniques to different application contexts. However, FT design in CPS should not be limited to enhancing detection, diagnosis, and recovery models through intelligent techniques

and algorithms. While these intelligent algorithms are often integrated into CPS using traditional approaches, namely, monolithic centralized or local approaches, this thesis argues that FT techniques in CPS must consider the system's ecosystem, where collaboration among different devices translates into both local and global benefits. By embracing collaborative FT, CPS can enhance the individual resilience of components and the overall robustness and adaptability of the system as a whole.

2.4 MULTI-AGENT SYSTEMS TO SUPPORT FT IN CPS

Effectively managing and deploying FT in CPS applications requires more intelligent and distributed software solutions. These solutions must exhibit high modularity, flexibility, reconfigurability, and adaptability while enabling dynamic interaction and collaboration within a heterogeneous and distributed environment. This Section introduces the concept of MAS and discusses why this paradigm is well-suited to supporting the implementation of FT in CPS.

2.4.1 Multi-Agent Systems Concept and Applications

The increasing demand for flexible, robust, and scalable FT solutions has driven the development of decentralized and distributed approaches that leverage CPS architectures. Distributed Artificial Intelligence (DAI) emerged within AI to address the challenges of designing intelligent solutions for these complex environments. One prominent example of DAI is the MAS paradigm (QIN *et al.*, 2016), which consists of independent, intelligent, and autonomous systems, referred to as agents, that collaborate and coordinate their activities based on local knowledge and individual capabilities to achieve common objectives (ABAR *et al.*, 2017). This paradigm provides a promising and effective design model for managing complex, heterogeneous, and distributed systems.

There is no consensus in the academic community regarding the definition of an agent, as the importance of specific attributes varies depending on the perspective of different experts. The literature presents multiple definitions, each emphasizing distinct characteristics. Some of these proposed definitions are summarized below:

- "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" (SINGH, 1998)

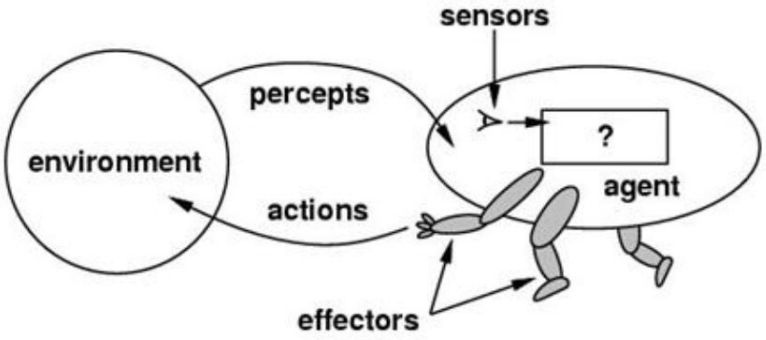
- “an agent is a computational entity that can be viewed as perceiving and acting upon its environment, that is autonomous and that operates flexibly and rationally in a variety of environmental circumstance” (WEISS, 1999)

- “an agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors” (RUSSELL, 2010)

- “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” (WOOLDRIDGE, 2009)

In general terms, regardless of the adopted definition, an agent is characterized by its ability to act and adapt flexibly, responding effectively to environmental changes through sensing. Additionally, an agent possesses autonomy, and its proactiveness enables it to take the initiative in pursuing its own goals (WOOLDRIDGE; JENNINGS, 1995). Figure 16 illustrates the main characteristics of an agent, highlighting its interaction with the environment and its reasoning mechanism, which determines the actions the agent should take based on its perceptions and internal states.

Figure 16 – Representation and mains mechanisms of an agent.



Source: (RUSSELL, 2010)

The behavior of an agent is determined according to a set of properties (WOOLDRIDGE, 2009; WOOLDRIDGE; JENNINGS, 1995),namely:

- Autonomy: agents operate without the direct intervention of humans, other agents or systems and have some kind of control over their actions and internal states;

- Social ability: agents interact with other agents (and human) to achieve their goals, through specific languages and communication protocols, which allow them to negotiate and cooperate rather than simply exchange information;
- Reactivity: ability to observe their environment and respond in a timely fashion way to changes that occur in it;
- Pro-activity: the ability to display goal-driven behaviors, taking the initiative to reach their goals, rather than simply act in response to the environment stimulus

In some cases, a single agent can be enough to control, monitor, or carry out the execution of tasks. Nevertheless, considering the context of this work, where multiple devices integrate a CPS environment, some tasks need to be performed by more than one agent due to the size and/or complexity of the system, appearing in this way, the concept of multi-agent systems. MAS is comprised of a society of agents that interact to achieve their goals and execute the task. Despite the necessity to meet its own goals, each agent can act on behalf of different users or perform tasks to meet the purposes of an application. To succeed in their interactions, they require cooperation, coordination, and collaboration capabilities (RUSSELL, 2010).

- Cooperation: the ability to work together to achieve common goals, when a single agent could not achieve;
- Coordination: the ability to manage the interdependence between activities, for example, to use non-shareable resources, or to perform a sequence of tasks;
- Collaboration: is characterized by an agent assisting another agent in executing specific tasks without expecting to have individual benefits.

With these characteristics, MAS has been applied in several areas of research and development to solve problems related to introducing intelligence and autonomy in different environments. Key areas with significant impact on I4.0 employ MAS as a flexible, robust, and dynamic solution, such as computer networks and IoT, robotics, complex systems, and smart grid. Related to computer networks and IoT, the author Queiroz *et al.* (2019) uses MAS to optimize the computer network setup to process industrial data evaluating the suitable approach between fog, edge, and cloud computing. MAS has been widely applied in robotics to manage and orchestrate the interaction between multiple robots (VOROTNIKOV *et al.*, 2018). The work

developed by Kalempa *et al.* (2020) the MAS solution is implemented in multiple mobile robots to obtain a consensus on allocating robot tasks, with priority requirements in a dynamic system. In complex dynamic systems, a centralized approach is costly and incurs significant processing overhead due to the demand for powerful modeling platforms and high complexity (DORRI *et al.*, 2018). In this research line, the work developed by Rodrigues *et al.* (2018) introduces an agent-based approach for service reconfiguration in manufacturing systems that dynamically identify possibilities and the on-the-fly implementation of new arrangements leading to better production efficiency. In the area of smart grid, MAS based solutions address challenges of generated and balanced energy, balancing between the energy consumer and producer over the energy price, storing energy in the home storage, and energy restoration (DORRI *et al.*, 2018). The review by Merabet *et al.* (2014) presents several MAS applications for these smart grid challenges.

In summary, the use of the MAS paradigm to model and implement CPS has gained considerable attention in the last few years, and the overviews in Karnouskos *et al.* (2020) reveal a significant amount of research devoted to applications of agents in CPS and industrial domains. Different solutions with centralized, hierarchical, hybrid, and distributed architectures are applied in production planning, manufacturing and process control, scheduling, and reconfiguration (LEITAO *et al.*, 2016).

2.4.2 Why Multi-Agent Systems to Address FT in CPS?

Agents differ from conventional approaches due to their inherent ability to autonomously adapt to changing conditions based on their internal state and environmental perception, without external intervention (LEITAO *et al.*, 2016; WOOLDRIDGE, 2009). In this context, the MAS approach provides an alternative paradigm for designing large-scale, intelligent, self-organizing, and adaptive systems with FT capabilities. This is achieved through decentralized control and decision-making, enabled by distributed, autonomous, and collaborative entities (PIARDI *et al.*, 2023). In such an approach, system behavior emerges from agent interactions, where decisions are made in a decentralized manner, contrasting with traditional monolithic, centralized, and rigid structures that lack flexibility, robustness, and on-the-fly reconfigurability (LEITÃO *et al.*, 2016; WANG *et al.*, 2016).

Taking into account these attributes and the intrinsic properties of software agents, MAS has been recognized as a suitable approach, providing the necessary abstractions and

mechanisms to support the design and development of CPS (HEHENBERGER *et al.*, 2016; KARNOUSKOS; LEITAO, 2016). For instance, agents can collaborate autonomously within distributed cyber-physical components (e.g., by embedding data analysis and decentralizing decision-making) while enabling system-wide dynamic adaptation (e.g., through collaboration protocols and self-organization strategies that facilitate both vertical and horizontal interaction and interoperability among components).

However, MAS-based CPS are inherently susceptible to faults occurring in individual agents, where an untreated fault in one agent may propagate to others, creating a domino effect. Nevertheless, the MAS paradigm provides a solid foundation for advancing FT in CPS, as it supports the development of diverse applications, is inherently decentralized and distributed, and integrates various technologies, such as AI, ML, IoT, M2M, cloud computing, and edge computing. In a distributed approach, each agent can detect faults in itself or identify faults in other agents and, through interaction and communication, notify other agents of conditions that compromise system availability. Additionally, MAS enables embedded data analysis techniques and agent-based processing, leveraging AI and ML as powerful tools for identifying fault behaviors and making recovery decisions without disrupting other tasks. These specialized software routines, executed in a distributed and autonomous manner, enhance system robustness by ensuring continued operation despite faults, thereby preventing a complete system failure.

Beyond academic research, several European research and development projects have reported the successful application of MAS in CPS domains, as discussed in Leitão *et al.* (2016), including SOCRADES, GRACE, IDEAS, IMC-AESOP, and ARUM. These studies, along with the growing interest in MAS for developing CPS solutions, particularly in the industrial domain, have been supported by the IEEE IES Technical Committee on Industrial Agents[4]. Despite this progress, several challenges still hinder the widespread adoption of MAS in real-world applications, particularly regarding dependability, reliability, and FT. The increasing adoption and importance of MAS-based CPS in I4.0 present new opportunities for further advancements in FT research within CPS.

MAS represents a fundamental technology for decentralized and collaborative systems. The autonomous and distributed nature of MAS enables efficient execution of complex tasks by distributing responsibilities among autonomous agents, allowing for process parallelization and mitigating bottlenecks commonly found in centralized or local systems, such as a single point of

[4] https://tcia.ieee-ies.org

failure. Moreover, agents' adaptive and self-organizing capabilities, through bioinspired behaviors or data-driven AI and ML techniques, enhance system robustness and resilience, allowing dynamic responses to environmental changes and localized faults. The information exchange and collaboration among agents, facilitated by communication and coordination mechanisms, improve decision-making in decentralized and distributed scenarios, such as sensor networks, intelligent logistics, and urban traffic management. These features contribute to more accurate fault detection and diagnosis. Therefore, adopting MAS not only enables scalable and flexible solutions but also enhances effectiveness and innovation in systems that require collaboration, decentralization, and resource distribution while maintaining robustness, dependability, and resilience.

2.5 CHALLENGES AND RESEARCH OPPORTUNITIES

As discussed in the previous Sections, integrating FT mechanisms into CPS architectures has significant potential across various domains. Applying new techniques and technologies can substantially enhance these systems' reliability, robustness, flexibility, and scalability. In this context, based on the literature review presented in this Chapter, several key challenges associated with implementing FT in CPS have been identified.

With this in mind, this Section will discuss these challenges from two perspectives. First, a broad perspective will be presented, covering general challenges relevant to the field, even though not all of them fall within the scope of this thesis. Subsequently, the discussion will focus on the specific challenges directly related to this research, highlighting how the proposed approach contributes to advancing the state of the art in the field.

2.5.1 Broad Perspective

To extract the entire potential from CPS applications, several research challenges arise in the context of FT and need to be addressed. Table 4 summarizes the key research challenges. These challenges are categorized into four distinct clusters.

In the engineering of individual FT nodes and components, the primary challenges revolve around developing each component's individual capabilities and behaviors, particularly their cyber and physical counterparts. This includes ensuring reliable and accurate data acquisition, aggregation, and management. The continuous data stream generated by various sensors

Table 4 – The main research challenges to develop FT in CPS.

Key Topics	Research Challenges
Engineering individual FT nodes and components	Efficient, reliable and real-time data acquisition, aggregation, and management
	Definition of the proper AI algorithms for FT and predictive prognostic
	Industry-oriented integration with physical assets (cyber-physical integration)
	Online optimization of FT models
Engineering of distributed and decentralized system with FT functionalities	Balancing and managing the distribution of intelligence among edge-cloud layers
	Definition of semantic data models for knowledge representation
	Support of data uncertainty, reliability, and inconsistency due to time and space myopia
	Definition of collaborative models to support autonomy and decision-make policies
	Industry-oriented communication for data exchanging (cyber-cyber interaction)
	Ensure security and data privacy mechanisms
Engineering of systems based on emergent behavior and self-FT	Controlling the system's emergent behavior in face a fault episode
	Definition of mechanisms that balances the continuous adaptation and disruptive reconfiguration
	Definition of boundaries for autonomic features (e.g., self-healing and self-adaptation)
	Control nervousness in self-organization processes
Infrastructures and methods	Low cost and constrained computational platforms to embed AI algorithms
	Communication infrastructures for fast, scalabe, robust, and overhead interactions
	Frameworks, testbed and benchmarks for different CPS domains
	Standards compliance

and actuators in CPS environments necessitates that nodes possess the capability to efficiently process and manage this data to maintain system integrity and performance. Consequently, robust data handling mechanisms must be developed to operate under strict time constraints tailored to the requirements of each application. Another critical aspect within this cluster is selecting and implementing appropriate AI and ML algorithms for fault handling and predictive prognostics. Decision-support systems for engineers and developers play a crucial role in guiding the correct selection and deployment of these algorithms according to specific application needs, thereby enhancing system robustness. Furthermore, seamless industry-oriented cyber-physical integration is essential to ensure the harmonious interaction between digital and physical components. This involves synchronizing software and hardware elements to create a cohesive system capable of dynamically responding to changes and faults.

The second cluster, engineering of distributed and decentralized systems with FT functionalities, focuses on the complexities of balancing and managing intelligence distribution across edge and cloud layers, which is crucial for scalable and efficient FT in CPS. A key challenge is defining semantic data models that facilitate effective knowledge representation, enabling consistent interpretation and utilization of data across diverse nodes. Additionally, addressing data uncertainty and inconsistency, often arising from temporal limitations and spatial myopia, is essential for maintaining system accuracy and trustworthiness. Collaborative models and approaches must be developed to support distributed decision-making, ensuring that nodes operate cohesively while adhering to established protocols. Furthermore, industry-oriented communication frameworks are required to enable seamless data exchange within the system's

cyber layer, while robust security and privacy mechanisms must be implemented to prevent cyber attacks and protect sensitive information. Collectively, these challenges emphasize the necessity of a holistic approach to designing distributed CPS that can operate reliably, autonomously, and securely in dynamic environments.

The third cluster highlights the challenges associated with engineering systems based on emergent behavior and self-FT. These challenges pertain to designing CPS that can autonomously adapt, reconfigure, and evolve in response to fault occurrences. A significant challenge is controlling the system's emergent behavior resulting from complex interactions between individual components. While emergent behavior can enhance adaptability and innovation, it also introduces risks of unpredictability, potentially compromising system stability and task execution. Therefore, it is crucial to define mechanisms that balance continuous adaptation with disruptive changes, ensuring long-term system functionality and reliability. Additionally, establishing clear boundaries for autonomic features, such as self-healing and self-adaptation, is necessary to prevent excessive automation, which could lead to unintended consequences or system instability. Another key issue is managing system nervousness in FT processes, where excessive reactivity to minor changes could destabilize operations. Addressing these challenges requires a refined approach that harnesses the benefits of self-organization while implementing safeguards to maintain control and predictability, ultimately enabling CPS to achieve both robustness and adaptability.

Finally, the fourth cluster, infrastructures and methods, addresses the foundational enablers required for implementing FT in CPS. Developing low-cost, resource-constrained computational platforms capable of embedding advanced FT and AI algorithms is a significant challenge. These platforms must balance computational efficiency with affordability, ensuring that FT approaches are both technically effective and economically viable, allowing for widespread adoption across various domains. Additionally, robust communication infrastructures enable fast, scalable, and reliable interactions across the system. These infrastructures must minimize overhead while maintaining high performance, even under network instability conditions. Another critical aspect is the development of frameworks, testbeds, and benchmarks tailored to different CPS domains, which are necessary for validating and comparing FT approaches in realistic scenarios. Finally, ensuring standards compliance is vital for interoperability and seamless integration across diverse systems, vendors, and industries.

Overcoming these challenges requires a multidisciplinary approach integrating ad-

vancements in hardware, software, and communication technologies, ultimately providing the necessary infrastructure to support resilient and efficient CPS.

2.5.2 Research Gaps Addressed

The research gaps that this thesis aims to address are directly related to some of the challenges previously discussed, namely: handling data uncertainty and inconsistency due to spatial myopia, cyber-cyber interactions, controlling the system's emergent behavior in the presence of faults, and defining mechanisms that balance continuous adaptation and disruptive reconfiguration. These aspects are integrated into the broader challenge of developing collaborative models to support autonomy and decision-making policies.

A key limitation in current approaches is the lack of approach that fully leverage the intrinsic capabilities of CPS, such as communication, interaction, and autonomous decision-making, to enhance FT while maintaining system robustness, scalability, and flexibility. Existing approaches tend to adopt either centralized strategies, where a single node is responsible for managing FT across the system. This, however, introduces drawbacks such as a single point of fault and limited scalability. Alternatively, some approaches rely on local strategies, where each component independently handles its own faults. However, these approaches suffer from restricted context awareness and limited computational resources, which can compromise the accuracy and effectiveness of FT mechanisms.

The lack of collaborative FT methodologies, in which components interact and share information to enhance FT, significantly constrains the adaptability and resilience of CPS. By leveraging the interconnected and interactive nature of CPS, a collaborative approach could substantially improve FT across all its stages, namely fault detection, diagnosis, and recovery.

The FT approach proposed in this thesis is grounded in the principles of MAS, serving as the backbone of the infrastructure. This approach highlights the components, agent behaviors, functionalities, and interaction protocols necessary to exploit communication and collaboration between agents, ultimately improving FT in CPS.

2.6 SUMMARY

This Chapter has explored the core concepts of FT and CPS, establishing the groundwork for understanding the thesis. Building on this foundation, a comprehensive literature characteri-

zation was conducted to examine the cross-relation of these two research topics to enhance the dependability and robustness of CPS by incorporating FT mechanisms. This analysis highlighted the growing academic interest in this topic, as evidenced by the evolution of publications over the years based on the Scopus database. Furthermore, an analysis of article abstracts was performed to identify the main attributes of FT, the main means of enhancing CPS robustness through FT, the key enabling technologies, and the CPS domains at the forefront of FT integration.

Subsequently, the related work outlined the principal approaches and methods of FT in CPS, providing a list of studies containing the current state of the art in this field. Emphasizing the importance of MAS in addressing FT challenges in CPS, the discussion delved into the main research challenges and opportunities, focusing on the specific gaps addressed in this work.

This Chapter has also highlighted that several existing works focus on specific stages of the FT process, often tailored to particular applications or CPS domains. Although some of these studies explore important requirements for FT in CPS, such as, leveraging IoT technologies for digitization, applying AI and ML for data analysis, or enabling components to autonomously tolerate or collaboratively manage faults, the works do not integrate these aspects into a comprehensive and generic approach, normally is focused in a specific application.

Moreover, combining these capabilities in a generic and modular manner, such that the approach can be instantiated across different CPS domains and application contexts, remains an open challenge. These findings directly respond to the motivation behind the first research question (***RQ-1***), which seeks to identify the requirements and features necessary to build a generic, scalable, and robust approach for FT in CPS.

By synthesizing these insights, this Chapter not only provides a thorough overview of the existing literature but also sets innovative contributions proposed in this thesis, aiming to advance the field of FT in CPS through an autonomous and collaborative approach.

3 COLLABORATIVE FAULT TOLERANCE FOR CYBER-PHYSICAL SYSTEMS

Based on the assumption that decentralized FT and collaborative capabilities are essential for developing robust, reliable, and resilient CPS, and given that MAS is a suitable technology for distributing intelligence, this thesis proposes a distributed and collaborative agent-based approach to enhance the FT capabilities of CPS in a generalized manner, thereby allowing the proposed approach to be instantiated across different CPS applications. A cyber-physical component refers to elements within the CPS that integrate computational capabilities (cyber component) with sensor and actuator (physical components), enabling interaction and control between the digital and physical environments (LEZOCHE; PANETTO, 2020). In the proposed approach, the cyber-physical components are embedded with intelligence, autonomy, and data analysis capabilities, further complemented by interaction and collaborative mechanisms. These components must contribute to the development of more robust, reliable, resilient, and adaptive CPS that can operate effectively in highly dynamic, heterogeneous, and open environments, such as those envisioned in the context of I4.0, particularly, achieving a high level of FT.

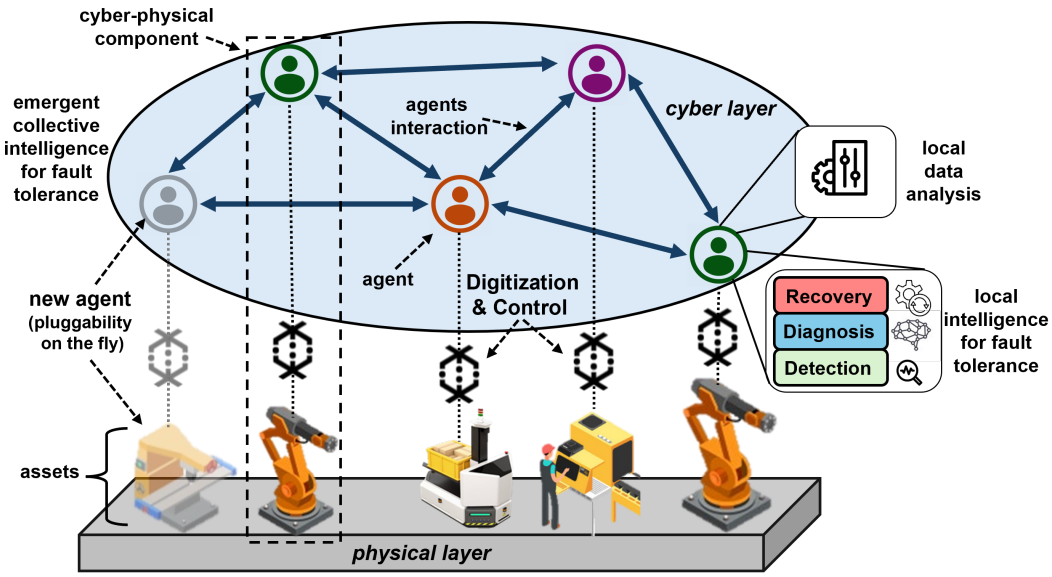
The remainder of this Chapter is structured as follows:

- Section 3.1 contextualizes and discusses the MAS-based CPS architecture, which serves as the foundation for collaborative FT.
- Section 3.2 introduces the concept of the CF* approach.
- Section 3.3 then presents and discusses the CF* meta-model, which outlines its implementation in the agents that constitute the CPS.
- Section 3.4 describes and models the collaborative stages of the CF*.
- Section 3.5 details the customized interaction protocols designed for each stage of CF*.
- Section 3.6 presents the Seven Collaboration Metrics (Seven Collaboration Metrics (7CM)), which evaluates the effectiveness of collaboration in fault detection and diagnosis stages.
- Finally, Section 3.7 summarizes the key aspects discussed in this Chapter.

3.1 MAS-BASED FAULT TOLERANCE ARCHITECTURE FOR CYBER-PHYSICAL SYSTEMS

The architecture to support FT within CPS incorporates the MAS to model the autonomous behavior of cyber components, facilitate decision-making, and enhance collaboration among cyber-physical components to improve the individual FT capabilities. This integration aims to achieve the system objectives while ensuring robustness and reliability of the CPS, as illustrated in Figure 17.

Figure 17 – FT within CPS architecture implemented with MAS technology.



The proposed collaborative FT approach operates in dynamic CPS environments, especially within the industry, where new operations emerge continuously, specific resources may become unavailable, and additional resources can be introduced at random intervals (ZHANG *et al.*, 2022). The occurrence of unexpected faults often causes deviations from initial and optimized plans, leading to reduced system performance. Under such circumstances, dynamic and agile response to faults is essential to maintaining adaptive, robust, and resilient CPS.

The architecture comprehends two layers, particularly the physical and cyber layers. For FT in CPS, each layer has different responsibilities, capabilities, functionalities, characteristics, and embedded features to enable the performance of the FT. The physical layer represents the physical systems or assets for which the FT is being employed to detect, diagnose, and recover from fault events. Apart from the system or assets, this layer accounts for the instrumentation system (e.g., sensors and actuators), which is responsible for implementing the action designated by the cyber component. Meanwhile, the cyber layer consists of software applications with

processing capabilities dedicated to executing computational tasks responsible for controlling and monitoring the physical asset and data analyzing for FT. Each cyber component locally possesses mechanisms to detect, diagnose, and recover from faults, in addition to the capability to exchange information with other system components to enhance its FT capacity. These two layers are integrated via a network through a digitization process that transmits information acquired by sensors in the physical layer to the cyber layer and vice-versa, transmitting decision-making information and parameter adjustments for asset operation. Consequently, integrating a physical asset with a software application, serving as an agent responsible for its control, monitoring, and fault tolerance processes, forms a cyber-physical component.

The proposed fault tolerance approach in CPS aligns with the MAS principles. Therefore, each cyber component, represented by an agent, exhibits autonomy, intelligence, and communication skills, enabling self-detection, self-diagnosis, and self-recovery at the local level. At the same time, enhanced capabilities of each stage emerge at the collaborative level through interactions among individual cyber components.

In the MAS-based FT architecture, individual agents are abstracting the physical assets, for instance, resources or products, and can perform each one of these functions locally by using properly embedded algorithms, providing fast response to condition changes, avoiding the central node dependency, overload, and ensuring control needs. However, since these individual agents have partial knowledge of the environment, they need to interact with each other to achieve an overall response to a fault episode, e.g., to detect, diagnose, or recover from the fault occurrence properly. This multilevel analysis, i.e., individual and collaborative levels, enhances the accuracy and performance of the detection and diagnosis processes and the fast adaptation and recovery to condition changes due to system disturbance.

The upcoming subsections describe in more detail the MAS-based FT architecture capabilities, including the features of the cyber-physical component with the cyber-physical interaction, the cyber-cyber interaction and information exchange, and in which computational layer the agents of this architecture can be deployed and executed.

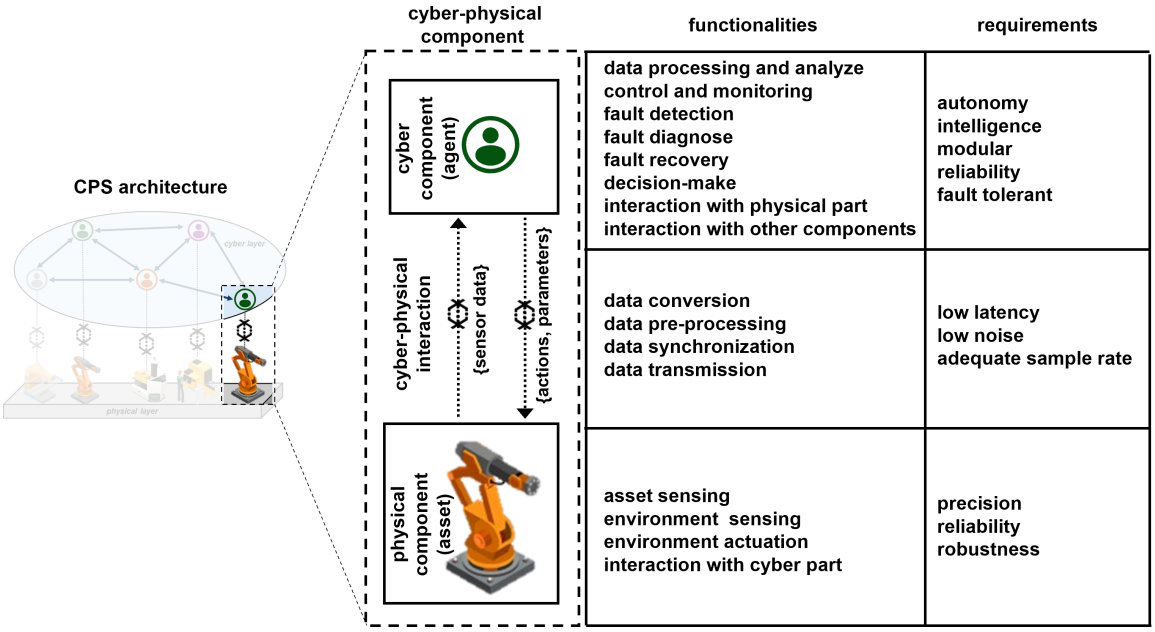
3.1.1 Cyber-Physical Component and Cyber-Physical Interaction

CPS consists of various interconnected cyber-physical components that work together to achieve the system goals as presented in Figure 17. The resilience and robustness of the CPS application implies maintaining the cyber-physical component operation at an acceptable

level of performance in response to disturbances, ensuring the reliability, safety, and efficiency of the system they form. The proposed approach integrates FT capabilities within the CPS architecture, leveraging the capacity to FT of cyber-physical components both individually and in a collaborative manner.

In the context of this work, the cyber-physical component, or just component, refers to elements that integrate computational processes with physical processes possessing communication networks with the ability to control and monitor assets and processing data to perform fault detection, diagnosis, and recovery to ensure that each cyber-physical component of the CPS possesses fault tolerance capabilities, as presented in Figure 18, relating each part of the cyber-physical component with its functionalities and requirements.

Figure 18 – Cyber-physical component for MAS-based FT in CPS.



The cyber-physical component is an autonomous, intelligent, and collaborative entity. It performs its designated operations (i.e., the expected tasks) based on algorithms and models instantiated according to its specific role in the CPS and its ability to interact with the environment. Similarly, in terms of FT, each cyber-physical component is equipped with local fault detection, diagnosis, and recovery models. These models, which can be implemented based on data-driven, model-based, or knowledge-based techniques, enable the component to continue executing its assigned tasks even in the presence of faults.

The physical component consists of the asset of each cyber-physical component. This asset contains sensors, actuators, and devices responsible for acquiring environmental data and interacting with the physical world. These assets are equipped with low-level controllers and

systems capable of digitizing and communicating with the cyber layer. The sensors ensure precise data acquisition, while actuators enable accurate interaction with the physical environment. The selection of sensors and actuators is based on the specific requirements of the CPS application.

The cyber component represents the computational elements of each cyber-physical component. It hosts dedicated software with robust processing capabilities and access to data generated by the physical components. For example, its computational responsibilities include high-level control, optimization of asset parameters, and performing condition monitoring and health management. To enable distributed and decentralized FT within the CPS architecture, avoiding reliance on centralized FT operations, the cyber component is equipped with fault detection, diagnosis, and recovery models. Additionally, the cyber component can interact with other cyber components for information and experience exchange, enhancing its FT capabilities. The adopted models and methods are highly adaptable and tailored to the characteristics of the cyber-physical component.

The MAS facilitates the engineering of cyber components within the CPS, modeling the autonomous behavior, decision-making, and interactions of these cyber components. The cyber components (i.e., agents) interact and collaborate to achieve shared objectives, such as optimizing production processes, managing resources, and decisions making in response to events. This collaboration fosters a more flexible, adaptive, and efficient system, culminating in collective intelligence. The exchange of information and collaboration among distributed components amplifies overall CPS intelligence, providing a holistic understanding of the system's state and dynamics (KARNOUSKOS *et al.*, 2020). In the proposed approach, the cyber component autonomy, intelligence, and collaboration intrinsic to the MAS is integrated into each cyber component modeled as an agent, incorporating models for fault detection, diagnosis, and recovery. Therefore, while agents demonstrate self-fault tolerance with local models, they collaborate to augment their capabilities when each agent cannot independently achieve the desired levels during any phase of FT.

The cyber-physical interaction is pivotal for digitizing the physical component and enabling seamless integration between the cyber and physical domains. The physical component transmits sensor data to the cyber component. In contrast, in the reverse flow, the cyber component sends commands to be executed by the physical component and parameters to be updated, such as parameter adjustments when the component is operating under fault conditions. This interaction can be implemented via wired connections, for instance, standardized interfaces such as Inter-

Integrated Circuit (I2C), Serial Peripheral Interface (SPI), analog and digital I/O ports, or through wireless connections using, for example, IoT technologies that adopt standards and protocols like Message Queuing Telemetry Transport (MQTT), Open Platform Communications Unified Architecture (OPC UA), and Constrained Application Protocol (CoAP) (LOMBARDI *et al.*, 2021). For legacy assets or older devices not initially designed for CPS integration, specialized adapters and the handling of proprietary communication protocols may be necessary to ensure effective interfacing with the cyber components.

Each component must be robust in the face of a fault, ensuring that it continues to operate, even in a degraded mode, after a disturbance, whether originating from that agent or an external agent within the system. To achieve this, it is essential that the agent can effectively detect, and diagnose a fault, and when possible recover from faults (JANUÁRIO *et al.*, 2019). The development of FT in this context requires that each cyber-physical exhibits the following features:

- **Autonomy:** Each agent operates autonomously, formulating decisions based on individual perceptions and fault tolerance models.
- **Decentralization:** The agents must operate without a central control or supervisory unit, thereby allowing them to make decisions derived from local observations and interactions with other agents of the CPS.
- **Adaptability:** Agents can evolve dynamically to optimize their performance in complex environments, enhancing their local ability to detect, diagnose, and recover from faults and improving their collaborative skills based on their execution experience.
- **Modularity:** Agents must support modularity, including dynamic service discovery, to facilitate integration with other agents within the system and enable adding new agents from a Plug-and-Play perspective (as illustrated in Figure 17). Regarding FT, modularity aims to ensure the customization of detection, diagnosis, and recovery mechanisms, allowing each agent to implement different FT modules tailored to the specific cyber-physical component. This characteristic directly contributes to the scalability and flexibility of the system, which are essential attributes for CPS.
- **Interoperability and Compatibility:** Agents should adhere to communication protocols and standards, enabling connectivity and information exchange among different com-

ponents, as the heterogeneity of devices and software is a characteristic of CPS. When multiple protocols or standards are required, brokers and adapters should be employed to ensure the interoperability and compatibility of the agents.

- **Collaboration:** Agents must coexist in a context where they can aggregate information to facilitate the execution of their service. In the context of fault tolerance, they should also be able to consolidate information to address a fault through interaction based on information and experience exchange. In this sense, an agent can be an entity that either requires or provides information and support through collaboration and interaction with other agents. Collaboration is characterized by an agent assisting another agent in executing specific tasks without expecting to have individual benefits.
- **Local Intelligence:** Agents must be empowered to make decisions and analyze their context based on data analysis and agent-based reasoning mechanisms. Additionally, intelligent algorithms within the realm of AI, utilizing descriptive, predictive, and prescriptive data analysis, such as ML, aim to extract patterns from data sets and develop models to support data-driven intelligence to mitigate faults.
- **Collective Intelligence:** Agents must be empowered by the intelligence resulting from collaboration among the local intelligence of agents to optimize decision-making and resource utilization. In terms of FT, this collaboration enhances the accuracy and effectiveness of models for fault detection, diagnosis, and recovery.

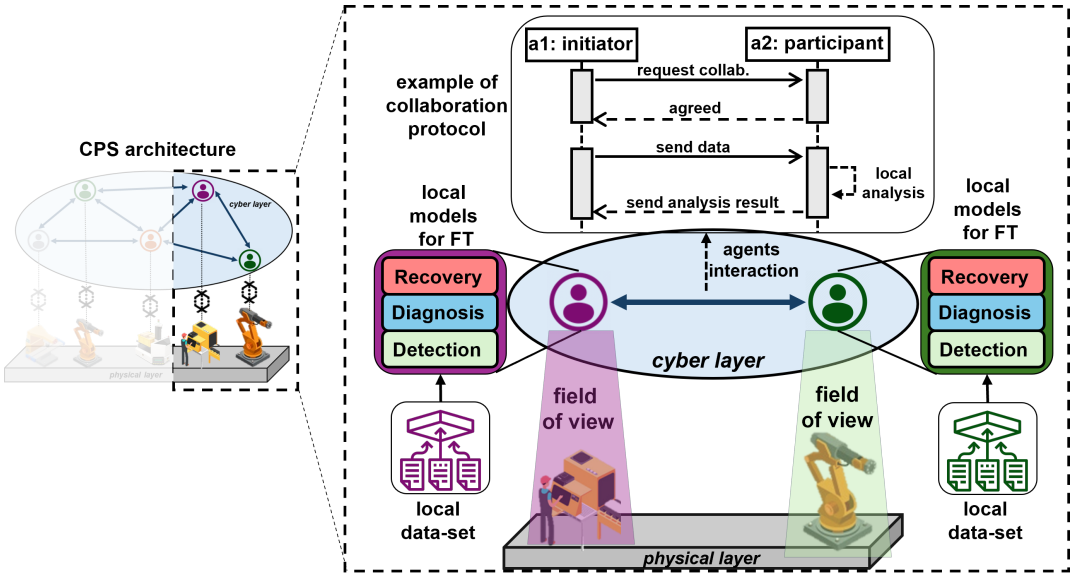
3.1.2 Cyber-to-Cyber Interaction

The cyber components need a means for communication and interaction in order to be able to collaborate and improve its FT capabilities. Therefore, besides the interaction between the agent and its corresponding physical component, the CPS architecture supports communication and interaction among agents, i.e., cyber-to-cyber interaction. This communication enables the exchange of information and knowledge between agents for various functionalities, including modular plug-and-play integration and service requests. In the context of FT, the interaction between agents can enable collaboration for fault detection, diagnosis, and recovery. This collaboration interaction consists of an agent requesting support from other agents to achieve a more robust, accurate, and sophisticated solution for fault detection, diagnosis, or recovery. The common goal of the collaborative MAS-based FT is to ensure that the CPS remains operational

even in the presence of faults. Each agent autonomously executes the FT mechanisms for its designated component while communicating and exchanging information with other agents to improve its FT capabilities. This collaboration enhances each agent's individual fault tolerance capabilities, contributing to the system's overall resilience. Through this collaborative process, agents can transcend their individual FT capabilities, thereby ensuring that fault occurrences do not fully compromise their operational efficiency and the system's overall functionality.

Figure 19 illustrates the interaction between two agents (although the proposed approach supports collaboration among multiple agents) within the CPS architecture. The CPS architecture has distributed FT, and each agent operates with local embedded models for fault detection, diagnosis, and recovery hosted within its cyber component. These models are tailored to the unique characteristics of each component, such as its historical dataset, which serves as a foundation for adapting the models to the specific operational context.

Figure 19 – Agent interaction for FT in the proposed CPS architecture.



However, agents operate with limited perspectives (i.e., myopia), as they rely solely on data streams from their respective physical components, restricting their "field of view" to local information. This inherent limitation highlights the importance of collaboration, which not only extends the informational context of an agent but also enhances the accuracy and reliability of its fault tolerance stages. Collaboration is triggered when an agent lacks sufficient confidence in its local solution and seeks support to improve the results of fault detection, diagnosis, or recovery while gaining a more comprehensive situational understanding.

Communication protocols are implemented to enable collaboration among agents for FT. These protocols define a set of standardized rules outlining the type, flow, and content

of information and messages, with predefined syntax and semantics shared among the agents involved. For the CPS architecture considered, the protocols for cyber-to-cyber interaction are based on Agent Communication Language (ACL), which provide a structured representation of message organization and utilize performative to define the intent and context of communication. For agent collaboration in the perspective of FT, these interaction protocols can derive their structure from established agent communication protocols, particularly those delineated by Foundation for Intelligent Physical Agents (FIPA), for instance, the FIPA Request[1], FIPA Query[2], FIPA Propose[3], among others. The structured agent interaction ensures effective communication to enhance the component FT capabilities.

Leveraging local intelligence and information exchange governed by interaction protocols to exchange information among cyber components enables collaboration in FT and represents an essential factor in the distribution of systems, providing scalability, flexibility, and robustness for CPS applications.

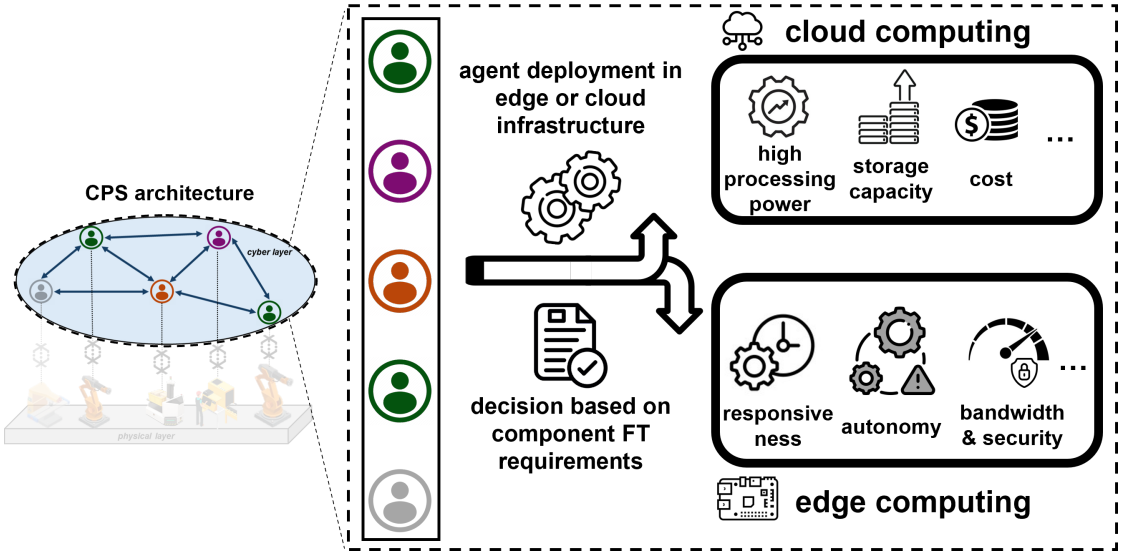
3.1.3 Edge Cloud Continuum for Agent Deployment

In the CPS architecture designed for distributed and collaborative FT, the deployment of agents plays a crucial role in determining the component's performance, scalability, robustness, and responsiveness. These components, which encompass functionalities such as fault detection, diagnosis, and recovery, and consequently are not lightweight software entities, considering the high-level and sophisticated reasoning capabilities and collaboration protocols, can be implemented to operate either in the cloud or at the edge network infrastructure. Each deployment paradigm offers distinct advantages and trade-offs, making the choice highly dependent on the specific requirements of the CPS application and the constraints of the operational environment as represented in Figure 20.

When deployed in the cloud, each agent benefits from vast computational resources with high processing power and storage capacity and the ability to perform large-scale fault analyses. This approach is particularly advantageous for tasks requiring extensive data aggregation and complex fault tolerance, mainly in the training and optimizing detection, diagnosis, and recovery models. Additionally, there is no requirement for concerns or investments related to the maintenance of the computing infrastructure. However, cloud-based deployment may

[1] FIPA Request: http://www.fipa.org/specs/fipa00026/SC00026H.html
[2] FIPA Query: http://www.fipa.org/specs/fipa00027/SC00027H.html
[3] FIPA Propose: http://www.fipa.org/specs/fipa00036/SC00036H.html

Figure 20 – Orchestration and concerns for deployment of MAS for CPS in cloud and edge continuum.



introduce latency, especially in safety-critical scenarios where responsiveness is essential and relies heavily on stable network connectivity along with cyber-security issues (REN *et al.*, 2017), which can pose challenges in some specific CPS applications, such as those found in power plant generation.

On the other hand, edge deployment allows the agents to operate closer to the physical components, analyzing data close to the device that collected the data, reducing latency, and enabling faster fault detection, diagnosis, and recovery. The agent deployment in the edge leverages local computational resources, enhancing responsiveness and reliability, particularly in scenarios where prompt action is required, ensuring faster fault detection and mitigation and minimizing downtime. Moreover, edge deployment enables greater autonomy for individual agents, allowing them to handle faults locally and collaborate effectively without excessive dependence on central infrastructure, and also enhance data privacy and security (CHEN *et al.*, 2018; CHIANG; ZHANG, 2016). However, adopting edge solutions implies using several distributed, heterogeneous, and constrained computing equipment (sometimes legacy systems), which may be complex and costly to deploy, manage, and maintain (HU *et al.*, 2017; SHI *et al.*, 2016).

Depending on their specific requirements and functional roles, the agents within the same CPS application can be deployed across different computational infrastructures, orchestrated by the edge and cloud continuum. This flexibility allows latency-critical to operate on the edge, while agents with computationally intensive or aggregated analyses are handled in the cloud. Despite being distributed across the cloud and edge, these agents maintain seamless

communication through cyber-to-cyber communication protocols. This capability ensures collaboration in FT, leveraging the strengths of each layer without compromising CPS integration or performance.

3.2 MAS-BASED COLLABORATIVE FAULT TOLERANCE

One of the central premises of this work is that each agent can exchange information, allowing each agent's knowledge, abilities, skills, and experiences to contribute to transcending individual capacity in FT. As observed in nature and even in the evolution of species, an individual can disseminate knowledge and acquire new insights through interaction and information exchange with other individuals. Similarly, the proposed approach aims to parameterize this behavior for FT in CPS.

Therefore, designing FT in CPS is not limited to improving models and algorithms. While optimizing and innovating techniques and algorithms for specific applications are essential, FT techniques in CPS must consider the ecosystem in which collaboration among different components translates into local and global benefits. This context enhances not only the individual capacities of each component but also the overall efficacy of the system.

The emergent collaborative FT behavior results from the synergistic collaboration among autonomous agents, wherein decision-making processes are distributed. This approach addresses the classic issues associated with traditional local or centralized monolithic fault tolerance approaches, such as scalability limitations, system myopia, reduced performance, and single points of fault.

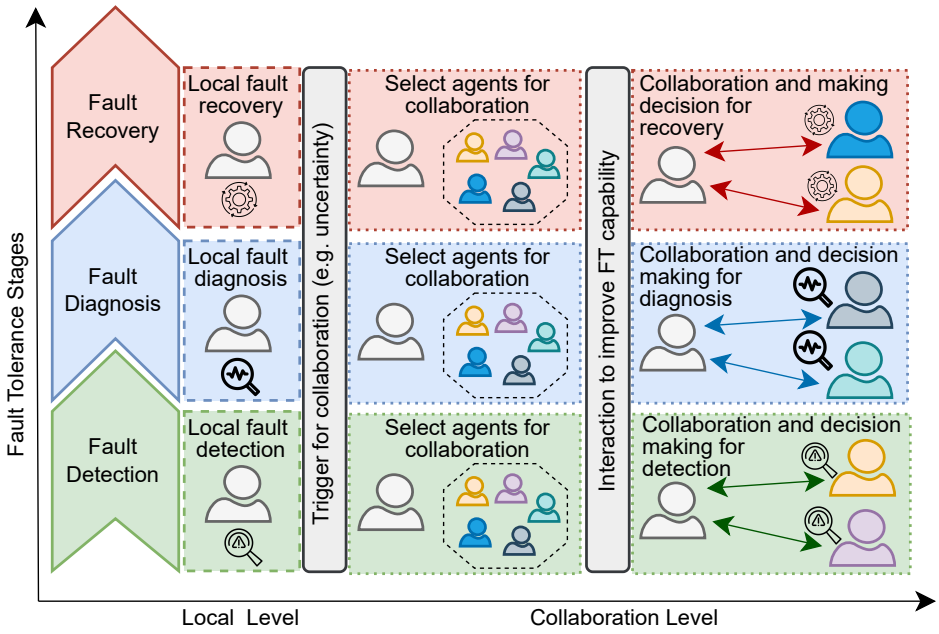
3.2.1 The CF* Approach

The CF* stands for Collaborative Fault *, a collective term that refers to the detection, diagnosis, and recovery stages of collaborative FT within the proposed approach. Specifically, the CF* encompasses the stages of Collaborative Fault Detection, Collaborative Fault Diagnosis, and Collaborative Fault Recovery, ultimately leading to Collaborative Fault Tolerance. The CF* introduces the self-fault tolerance capability in cyber-physical components and adopts collaborative behavior when appropriate to enhance the component FT.

Figure 21 provides a comprehensive overview of the CF* stages and the collaboration process. The fault tolerance capability of the CF* approach manifests on two levels: local

and collaborative, as shown in Figure 21. Each agent is equipped with specialized models corresponding to each stage of fault tolerance. Specifically, there is a distinct mechanism or algorithm designated for the detection, a separate one for the diagnosis, and another for the recovery. When a fault is well known and aligns with the agent's detection, diagnosis, and recovery models, the local level is sufficient to handle the disturbance. However, when the agent's models cannot address the fault independently, the collaborative level is activated, utilizing the experience and information from compatible agents to improve the capability to manage the episode and find a viable solution.

Figure 21 – Overview of collaborative process for MAS-based FT for CPS.



All stages occur within individual agents at the local level, which executes self-fault tolerance processes to respond quickly to disturbances and minimize the fault impact. For instance, an agent representing a mobile robot can detect a disturbance in movement speed, then diagnose the issue as a problem with the controller gains, and recover by adjusting these gains to achieve the desired speed and movement for the robot operation.

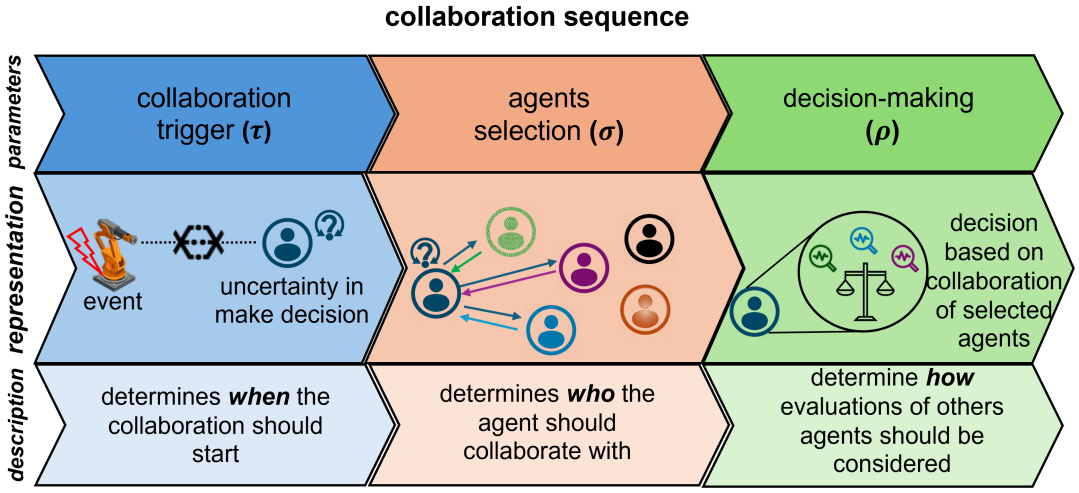
The collaborative level of fault tolerance is activated when an agent cannot individually perform one or more fault tolerance stages. This level, as an innovative feature, enhances decision-making, particularly in complex CPS. In such cases, a specific collaboration process is initiated for the required stage, with the initiator agent requesting assistance from other compatible agents based on the needed stage. For example, if uncertainty arises during the detection stage, the agent collaborates with other agents' detection stages. This principle applies similarly to the

diagnosis and recovery stages. Multiple stages may require collaboration for the same event; for instance, detection might be handled locally, while diagnosis and recovery involve collaboration. Decisions regarding detection, diagnosis, or recovery are facilitated through agent collaboration, enabled by interaction and data exchange.

3.2.2 Collaborative Parameters for the CF*

For each agent to be self-fault tolerant and to enhance its FT through collaboration when appropriate, the CF* introduces three collaboration parameters: τ, σ, and ρ, as illustrated in Figure 22. These parameters are embedded within each agent and serve to define the critical aspects of collaboration, specifically *when*, *who*, and *how*, the collaboration in the fault detection, diagnosis, and recovery stages should take place. The purpose of collaboration is to transcend the individual agent's capacity to detect, diagnose, and recover from faults by leveraging the knowledge, data, and experience of other compatible agents within the system. In practice, the instantiation of these parameters allows tuning each agent's collaborative behavior to fit the requirements of the CPS application. Consequently, the responsibility of deploying CF* lies with the system designers and engineers, who must tailor the collaboration parameters to the specific characteristics and goals of the system, ensuring that the desired level of fault tolerance and collaboration behavior is achieved.

Figure 22 – Representation and description of the collaboration parameters in CF*.



By appropriately configuring these collaboration parameters, the agents can dynamically adapt their behavior to the specific needs and context of the CPS. The collaboration trigger parameter, denoted as τ, defines the conditions under which an agent determines the need for

collaborative support to make a decision. When critical thresholds or uncertainties are met, the agent recognizes that it may not be able to effectively detect, diagnose, or recover from a fault independently, prompting it to engage in collaboration. This parameter ensures that collaboration is initiated when necessary.

Once collaboration is triggered, the next step is identifying which agents will participate in the information exchange. This process, known as agent selection, is governed by the parameter σ, which ensures the selection of the most appropriate agents for collaboration. This dynamic selection process ensures that the initiating agent identifies the most relevant peers and the number of participants that can contribute to collaborative fault detection, diagnosis, or recovery, thereby preventing unnecessary overload and optimizing resource utilization.

After selecting and interacting with other agents, the initiator agent must analyze the data and information obtained from the collaboration to make a decision. This decision-making process is governed by the parameter ρ, which plays a critical role in guiding how the agent processes the feedback and contextual information gathered from its peers. The ρ parameter allows the agent to adapt its strategy based on the collaborative input, enhancing the effectiveness of fault tolerance at any stage, thus improving the overall resilience of the CPS.

Table 5 presents examples of potential strategies that can be instantiated for each collaboration parameter, illustrating when, who, and how the collaboration process should occur. The correct instantiation of these parameters directly influences the collaborative behavior of each agent. Moreover, the flexibility of these strategies is a crucial factor, as they must be tailored to the specific characteristics of each agent and the particular context of the CPS application. Table 5 offers examples for each parameter that can be adapted and customized for different scenarios, thereby enhancing the effectiveness of the CF* approach. The collaboration can yield more robust and efficient outcomes by fine-tuning these parameters.

3.3 CF* META-MODEL FOR A GENERIC AGENT

According to the CF* approach, each agent operates its FT independently and can collaborate when needed, enhancing FT through established interaction protocols and leading to collaborative FT. The customization of FT algorithms and techniques within each agent can be tailored to specific applications and incorporate data-driven, model-based, or knowledge-based models, including statistical or AI approaches. This approach ensures scalability, adaptability, and robustness, offering a comprehensive and resilient solution for CPS applications where

Table 5 – Examples of instantiation strategies for the collaborations parameters covering all FT stages.

Parameter	Instantiation Strategies	Strategy Description and Examples
Trigger τ	uncertainty or inconclusiveness	occurs when data analysis results in uncertainty for some stage of FT, e.g., low similarity in inference for making a decision in detection, diagnosis, or recovery.
	collaboration defined	collaboration pre-defined, e.g., time-out or after a certain amount of inferences.
	limited perception	case the agent's information is insufficient to make a decision, collaboration begins, for example, system information myopia.
	limited resources	occurs when the agent faces a computational overhead, e.g., processing overhead for executing the detection, diagnosis, or recovery models.
Selection σ	compatibility	selects the most compatible agents for collaboration
	proximity	selects agents that are physically near.
	computational capacity	selects agents with the greatest processing power.
	availability	selects agents that are idle
Decision-Making ρ	most voted	decisions based on the majority of votes obtained from other agents
	weighted decision	decision based on the weighted votes of other agents
	more relevance	decision made by the agent most confident in the collaboration

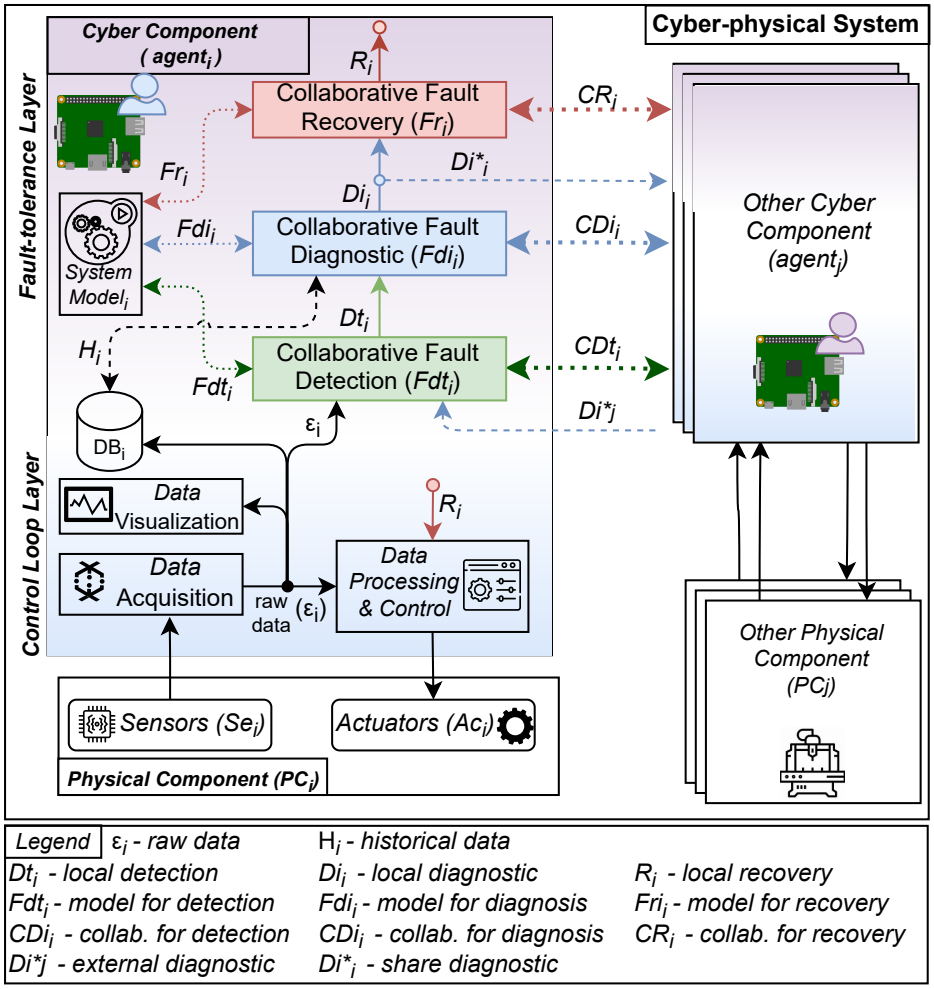
reliability and precision are paramount.

Figure 23 illustrates the CF* meta-model, which outlines the internal modules (i.e., functional units designed to perform specific tasks, encapsulating distinct functionalities) for an individual agent, as well as the stages of FT, highlighting the potential collaboration between these stages across different agents.

Individually, the agent comprises essential modules designed for data analysis, used to self-control the component, organized within the control loop layer, as well as for executing the stages of fault tolerance, grouped within the fault tolerance layer of the CF* meta-model (see Figure 23). Each module within the agent is self-contained, meaning it performs atomic functions independently and operates as an agent behavior, enabling the design and customization of different agent configurations and simplifying both the agent's development and deployment. This adaptability allows for tailoring the specifications of each module according to the requirements and constraints of the cyber component and designing fault tolerance models to address the various disturbances to which the component may be susceptible. These principles allow agents to be adapted and configured to operate across different components. The internal arrows of the agent in Figure 23 represent data flow between the modules. However, depending on the specific application, it may be necessary to remove certain proposed modules or incorporate additional ones not considered in this meta-model, such as auto-encoders and auto-decoders, to optimize the size of information exchanged between agents.

The physical component (i.e., the asset) controlled by the agent with CF* behavior comprises modules responsible for sensing, digitization, and the ability to act upon and modify

Figure 23 – CF* meta-model representing the internal modules of cyber physical components and the collaborative FT stages.



its environment. Therefore, the physical component PC_i of the $agent_i$ [4] contains a set of sensors Se_i for data acquisition and a set of actuators Ac_i for environmental interaction. For instance, considering $agent_i$ instantiate for an Autonomous Mobile Robot (AMR), the sensor set Se_i may include battery level sensors, encoders and accelerometers to monitor wheel and robot movement, ultrasonic and lidar sensors for obstacle detection, and cameras for frame capture to enhance the AMR navigation. In the AMR example, the actuator set Ac_i may include motors for robot movement. The physical component transmits the asset raw data (ϵ_i) to $agent_i$ and receives information and commands to interact with the environment, as illustrated in Figure 23.

Although not the central focus of this work, the modules within the control loop layer are essential in supporting the agent's continuous operation by supporting the digitization, processing, and control functions. The *Data Acquisition*, *Data Processing & Control*, and *Data Visualization* modules represent the digitization of data used to execute local control of the PC_i,

[4] The index i designate local agent.

sending commands to the set of actuators (Ac_i) to reproduce the expected behavior. The function of these modules is to perform operational tasks related to monitoring and control, aiming to meet fast response requirements and ensure the operational autonomy of the component, i.e., without dependence on centralized monitoring and control nodes. Additionally, given their proximity to the physical components and their respective data sources, the control loop modules can continuously process and analyze raw data (ϵ_i) to meet the control requirements of the PC_i. These modules that constitute the control loop layer are detailed in Appendix B to preserve the central narrative of this work on the collaborative fault tolerance, i.e., on the modules pertinent to the fault tolerance layer of the CF* meta-model.

3.4 MODULES OF CF* FAULT TOLERANCE LAYER

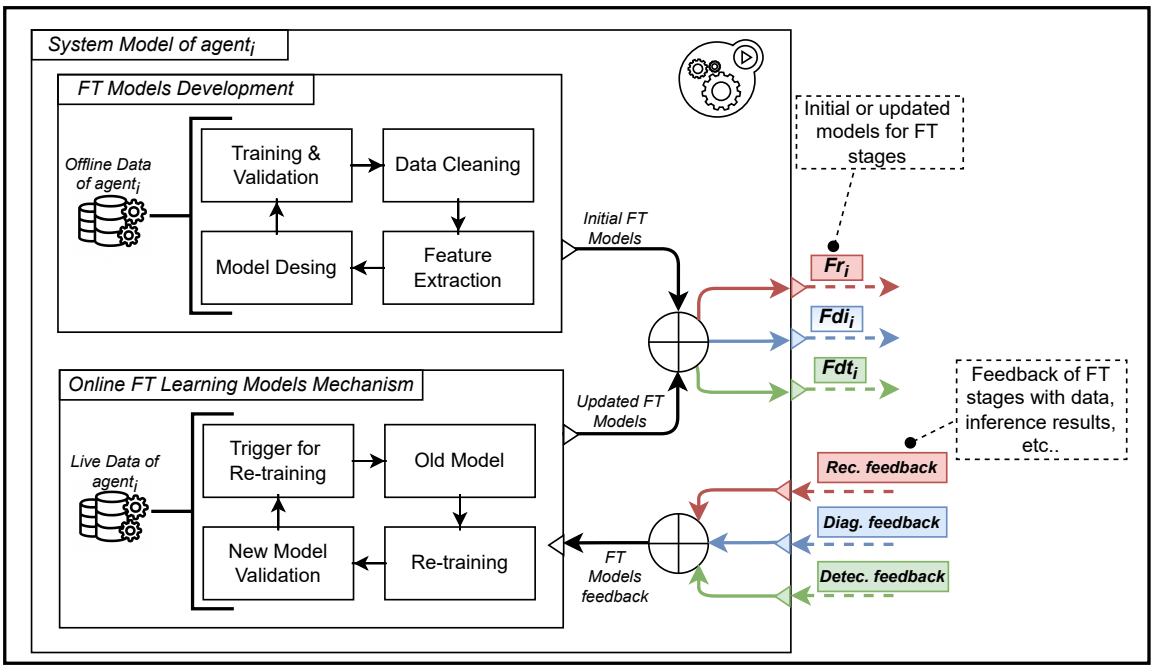
The fault tolerance layer, as represented in the CF* meta-model, illustrated in Figure 23, enhances the resilience and operational availability of the component in the fault events. From this point forward, this work will focus on the modules constituting the fault tolerance layer within the CF* meta-model. These modules, specifically the *system model*, *collaborative fault detection*, *collaborative fault diagnosis*, and *collaborative fault recovery* are essential for ensuring the robust and reliable operation of the components, even in the presence of disturbances and adverse conditions. The customization, by CPS designer and engineer of detection, diagnosis, and recovery algorithms and techniques within the fault tolerance layer can be tailored to specific applications and incorporate data-driven, model-based, or knowledge-based models, including statistical or AI algorithms.

3.4.1 System Model

The *system model* module is essential for each $agent_i$ within the context of the CF* approach, with the responsibility of developing and managing the FT models for each FT stage. The function of the system model is to develop, orchestrate, and improve the models for fault detection (Fdt_i), fault diagnosis (Fdi_i), and fault recovery (Fr_i) specific to each $agent_i$ as presented in Figure 24. These models can be uniquely tailored for each agent, distinguished by factors such as different AI algorithms, parameters, and training datasets, ensuring that they operate cohesively and contextually within the CPS application and the components.

Upon the meticulous development of the initial models, encompassing design, data

Figure 24 – Structure of the system model of $agent_i$.



cleaning, feature extraction, training, and validating the models, the system model acts in the continuous enhancement of these models. As $agent_i$ operates, the $SystemModel_i$ collects feedback from the fault tolerance stages. This feedback comprises information on the effectiveness of the detections performed, the accuracy of the diagnoses, the success of the recovery actions, and insights from collaboration. This feedback loop is instrumental in refining the underlying models, enabling the system to adapt and enhance its capabilities as new data is processed and collaborative interactions occur. Therefore, when confronted with a reduction or degradation in the performance of any FT model, such as those resulting from changes in the operational context of a component, the system model can retrain and subsequently update the model. Consequently, the system model ensures that the fault tolerance mechanisms are not static but instead dynamic and responsive to the operational environment, thereby improving the overall reliability and robustness of the CPS.

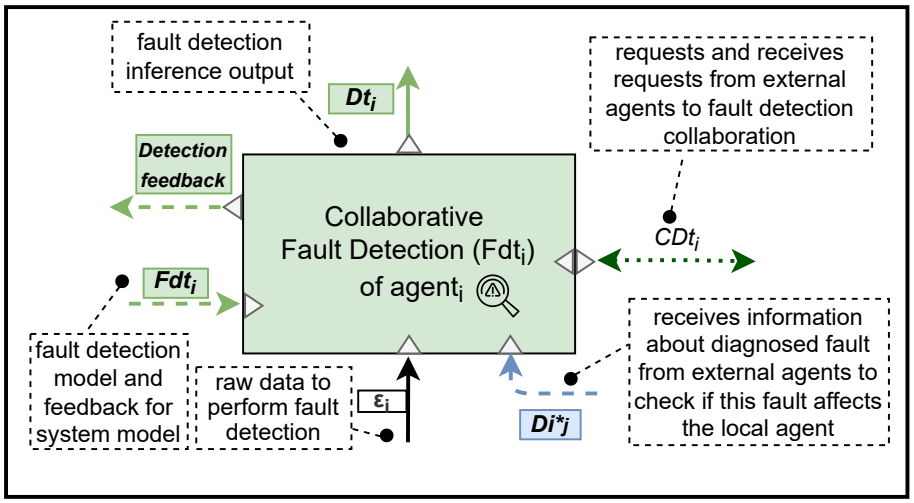
In cases where the $agent_i$ is deployed in the edge computing infrastructure, for instance, to be computationally close to the asset, reducing the FT latency, and the computational resources available are insufficient to support the operations of this module, the module deployment can be dismembered to a computational layer with higher capacity, for example, the cloud computing. This infrastructure provides enhanced processing power and storage capabilities, enabling the efficient development and improvement of the FT models. However, ensuring that the interaction between the *system model* and $agent_i$ remains transparent and synchronized is essential.

3.4.2 Collaborative Fault Detection

The fault detection stage is fundamental for the component, serving as the initial stage in the FT process. In the detection module, the fault occurrences in the component are identified, in other words, this module determines the occurrence or absence of a fault. Faults should be detected as soon as they occur to avoid considerable impact; therefore, the models and algorithms responsible for detection should not be computationally demanding. Considering these aspects, detection must minimize false alarms and undetected faults.

As illustrated in Figure 25, utilizing detection models and algorithms provided by the system model, this module analyzes the raw data (ϵ_i) to promptly detect any disturbances or malfunctions that could compromise the operation of the $agent_i$ or potentially of the CPS.

Figure 25 – Description of collaborative fault detection module for $agent_i$.



Moreover, according to the collaboration parameters, this module can request or receive collaboration requests from fault detection modules of other agents within the system. This collaborative approach is particularly beneficial in critical situations, such as when there is uncertainty in detection (as illustrated by the CDt_i in Figure 23 and Figure 25). By leveraging the collaborative capabilities of multiple agents, the components enhance their fault detection accuracy.

Furthermore, a fault diagnosed by an external agent (e.g., $agent_j$) may affect the local $agent_i$. Thus, as illustrated in Figure 25, the diagnosis provided by the external agent Di^*_j is subjected to analysis by the fault detection model to verify its impact on the $agent_i$ operation.

With this in mind, consider the set $\mathcal{A}$ of n agents, where each $agent_i$ has local fault detection models and collaborative capabilities. The fault detection output (Dt_i) of $agent_i$ is

generated when the fault detection model (Fdt_i) identifies a fault in the raw data (ϵ_i) obtained from the *Data Acquisition* module or when receiving a diagnosis (Di_j^*) from external $agent_j$. Therefore, it is possible to express the detection output Dt_i as a function of fault detection model (Fdt_i) as described by the Equation (1):

$$Dt_i = Fdt_i(\epsilon_i, CDt_i(\tau,\sigma,\rho)) + Di_j^* \rightarrow \{pdt_i, \varphi_i\} \mid \begin{cases} i \in \mathbb{N}^* \mid \mathbb{N}^* = \{1,...,n\} \\ j \in \mathbb{N}^* \mid \mathbb{N}^* = \{1,...,n\}, j \neq i \\ pdt_i \in [0,1] \\ \varphi_i \subset \epsilon_i \end{cases} \quad (1)$$

where ϵ_i represents the raw data observation from the set of sensor Se_i and CDt_i denotes the collaboration model, based on the collaboration parameters, used when $agent_i$ meet the collaboration trigger criteria, e.g., has uncertain about classifying the observation ϵ_i as a fault. Finally, fault detection considers the fault diagnosis Di_j^* from another $agent_j$, which may impact $agent_i$. The image of collaborative fault detection Fdt_i model (i.e., Dt_i output) is a tuple $\{pdt_i,\varphi_i\}$, where pdt_i represents the level of confidence related to the fault detection by $agent_i$ in the interval between $[0,1]$ as:

- pdt_i = 0 indicates that no fault was detected.
- pdt_i = 1 fault has been confirmed detected.
- $0 < pdt_i < 1$ represents confidence levels related to a fault.

The φ_i represents the data associated with the detected fault episode and constitutes a portion of the raw data ϵ_i. This detection output Dt_i in the form of a tuple $\{pdt_i,\varphi_i\}$ will be useful for the fault diagnosis stage to start its operation and be able to determine the fault, its nature, cause, and impact.

Table 6 summarizes the main functional features of the fault detection stage.

Upon detecting a fault, the collaborative fault detection module transmits relevant information to the diagnosis stage for the fault tolerance process. This transition ensures that the overall system maintains a high level of reliability and can quickly address emerging issues.

Table 6 – Collaborative fault detection stage features.

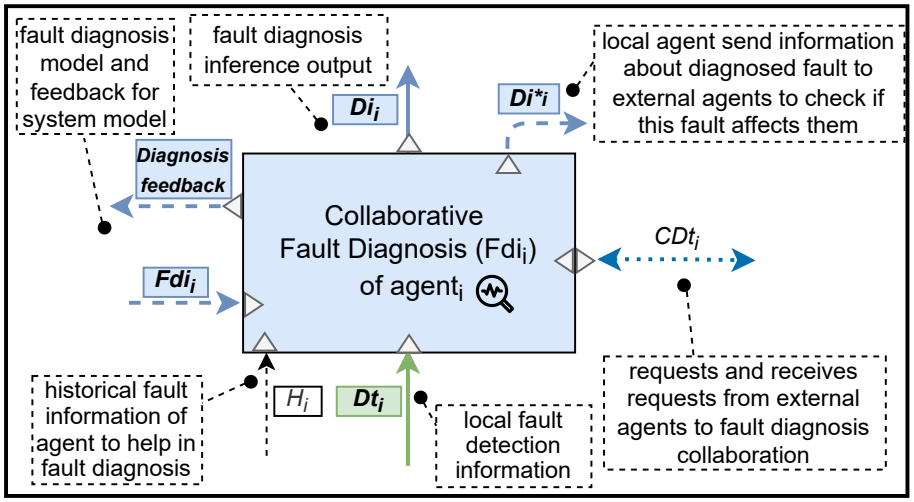
Feature	Description
Role	detect faults in the agent and minimize false alarms and undetected fault
Local input	data sample | detection model | collaboration model
Local output	fault detected | fault free | detection confidence | feedback for system model
Local capabilities	process data for fault detection
External input	receive request to collaborate in fault detection of external agent | receive fault diagnosis information of external agent
External output	responds to the collaboration request for detection | responds to whether the external fault affects the agent
Collab. capabilities	request collaboration to improve local capabilities | responds to collaboration requests to assist external agents in fault detection

3.4.3 Collaborative Fault Diagnosis

The *collaborative fault diagnosis* module represents the second stage in the FT process of a CF*. The objective of this stage is to accurately identify the root cause of the detected fault by determining its nature, origin, and impact, thereby providing the classification of the fault type, for example, determining which sensor or actuator failed.

As depicted in Figure 26, this module operates by leveraging the diagnosis models and algorithms supplied by the system model. It processes information obtained from the detection stage (Dt_i) while incorporating historical data (H_i) that contains detailed records of past fault occurrences, including observed symptoms, causes, and recurring fault patterns under specific conditions, enabling a thorough and precise fault diagnosis. Furthermore, this module aims to reduce the occurrence of diagnosis misclassifications. Compared with the detection stage, the diagnosis employs more sophisticated models that require more computational resources.

Figure 26 – Description of collaborative fault diagnosis module for $agent_i$.



Collaboration can occur in two ways for the diagnosis stage. The first involves cases where a conclusive local diagnosis cannot be achieved according to the collaboration parameters presented in section 3.2.2. In such instances, the module may request collaboration from the fault diagnosis modules of other agents within the system to transcend its individual capabilities (as illustrated by the CDi_i). This collaborative effort enhances the overall diagnosis accuracy and effectiveness. The second form of collaboration takes place once the diagnosis is finalized. In this case, the module disseminates the diagnosis information to the detection modules of other agents, as shown in Figure 23 with the arrows Di_i^*. This dissemination aims to assess whether the identified fault may impact other agents within the system, facilitating a comprehensive understanding of the fault's implications.

In this context, considering a set $\mathcal{A}$ of n agents, where each $agent_i$ has a local fault diagnosis model and collaborative capabilities, the system can diagnose a range of faults that the $agent_i$ might encounter during operation. As represented in the CF* meta-model in Figure 23, the fault diagnosis output (Di_i) and (Di_i^*) of $agent_i$ is generated when the fault diagnosis model (Fdi_i) performs its function based on the detection output information (Dt_i), the set of pre-mapped fault classes $\mathcal{K}_i$ that $agent_i$ is susceptible to, the fault history set ($\mathcal{H}_i$) of $agent_i$, and the collaborative fault diagnosis model (CDi_i), as formalized by Equation (2):

$$Di_i = Di_i^* = Fdi_i(Dt_i,\mathcal{K}_i,\mathcal{H}_i, CDi_i(\tau,\sigma,\rho)) \rightarrow \{f_x,\mathcal{SI}_i,\mathcal{CA}_i,\mathcal{IM}_i\} \tag{2}$$

where the domain of the fault diagnosis model (Fdi_i) is composed of:

- Dt_i: the fault detection output constituted by the tuple $\{pdt_i,\varphi_i\}$ which respectively represents the confidence range in fault detection and the subset of data with associated fault.
- $\mathcal{K}_i$: set of pre-mapped fault classes $\mathcal{K}_i = \{f_1, f_2, ..., f_k\} \mid \{k \in \mathbb{N}^*\}$ that the fault diagnosis model Fdi_i is suitable for classifying for $agent_i$. The element f_k represents the k-th possible fault class for $agent_i$
- $\mathcal{H}_i$: a set of historical faults f_k, characterized by their association with $agent_i$, which is useful for correlating faults during the diagnosis stage. $\mathcal{H}_i = \{hf_1,hf_2,...hf_k\} \mid \{k \in \mathbb{N}^*\}$, where the k-th element hf represent the history of faults for $agent_i$, with characteristics, cause and impact.

- $CDi_i(\tau,\sigma,\rho)$: denotes the collaboration model when the $agent_i$ meet the collaboration trigger criteria.

The diagnosis model Fdi_i can be based on model-based, data-driven, or knowledge-based mechanisms, and after the inference, it will produce an image of the fault diagnosis, i.e., the diagnosis output Di_t for $agent_i$, according to:

- f_x: fault that the diagnosis model (Fdi_i) indicates that is occurring in $agent_i$ due to observation of the fault data φ_i, ensuring that $\{f_x \in \mathcal{K}_i \mid 1 \leq x \leq k\}$.

- $\mathcal{SI}_i$: set composed of the similarity scores for all k fault classes, i.e., $\mathcal{SI}_i = \{sf_1, sf_2, ..., sf_k\} \mid \{sf_x \in \mathcal{SI}_i\}$. Each sf_k represents the similarity value indicating how closely the diagnosis of the fault associated with observation φ_i matches fault f_k. Each sf_k belongs to the range $[0,1]$, where 0 indicates no similarity, and 1 indicates maximum similarity. Furthermore, the diagnosed fault f_x is the one whose similarity value sf_x is the highest element in the set $\mathcal{SI}_i$.

- $\mathcal{CA}_i$: this set represents the potential causes linked to the diagnosed fault f_x. For each identified fault, understanding the underlying cause is crucial for accurate and effective fault recovery. The set $\mathcal{CA}_i$ encompasses all plausible root causes that could have triggered the fault f_x, providing valuable insights into the fault's origin, whether it stems from sensor degradation, actuator malfunction, communication errors, or other system disturbances.

- $\mathcal{IM}_i$: this set represents the possible impacts associated with the diagnosed fault f_x, e.g., $\mathcal{IM}_i = [globalfault, localfault, ...]$. The impacts can range from local issues affecting only the $agent_i$ to global faults that compromise a part or the entire operation of the CPS. Understanding the potential scope of the fault's impact is essential for determining the extension of the recovery actions. For example, a local fault may require minimal intervention to restore standard functionality, whereas a global fault could necessitate a system-wide response to prevent further cascading faults. By identifying the impact of the fault, $\mathcal{IM}_i$ provides critical information for prioritizing resources and selecting the most effective recovery strategies, ensuring that the agent can respond appropriately to maintain overall system stability.

Suppose $agent_i$ cannot classify the fault, for instance, due to uncertainties or equal similarity scores associated with different faults. In that case, the collaboration process for

detecting CDi_i is initiated, where parameters τ, σ, and ρ are used according to the idea presented in the collaboration parameters and the detection stage. After the complete process of diagnosing the fault, the fault diagnosis output (Di_i) sent a report to the recovery stage of the local agent and to the detection stages of other $agent_j$, as represented by Di_i^*, to assess whether the diagnosed fault impacts them. For example, when a mobile robot fails and congests a waypoint, it can alert other mobile robots that it failed at the particular waypoint. Consequently, the robots that receive this information can evaluate whether the fault impacts them and, if needed, execute a new path-planning procedure to avoid congestion.

Table 7 summarizes the main functional features of the fault diagnosis stage.

Table 7 – Collaborative fault diagnosis stage features.

Feature	**Description**
Role	diagnosing the fault, classifying, determining its root cause and impact
Local input	detection information ǀ historical data ǀ collaboration model.
Local output	fault class and similarity ǀ fault cause ǀ fault impact ǀ feedback to system model
Local capabilities	diagnose the disturbance that caused the fault
External input	receive a request to collaborate in the fault diagnosis of an external agent ǀ receives information if the fault affects other agents
External output	responds to the collaboration request for diagnosis ǀ shares the diagnosis with other agents
Collab. capabilities	request collaboration to improve local capabilities ǀ responds to collaboration requests to assist external agents in fault diagnosis

Upon completing the fault diagnosis and knowing whether the impact is local or broader, this module transmits relevant information to the next stage in the fault tolerance process, namely the recovery stage.

3.4.4 Collaborative Fault Recovery

The fault recovery module aims to restore the component functionality while minimizing the impacts caused by the fault event. It also seeks to mitigate or even correct the effects of the fault and prevent its propagation, which could compromise other agents within the CPS.

As illustrated in Figure 27, employing recovery models and algorithms supplied by the system model, this module operates based on information obtained from the diagnosis stage (Di_i). The recovery module provides an action recovery plan enabling the $agent_i$ to maintain operational functionality despite encountering a fault episode.

With this in mind, consider a set $\mathcal{A}$ of n agents, where each $agent_i$ possesses local fault recovery models and collaborative functionalities. As illustrated in the CF* meta-model in

Figure 27 – Description of collaborative fault recovery module for $agent_i$.

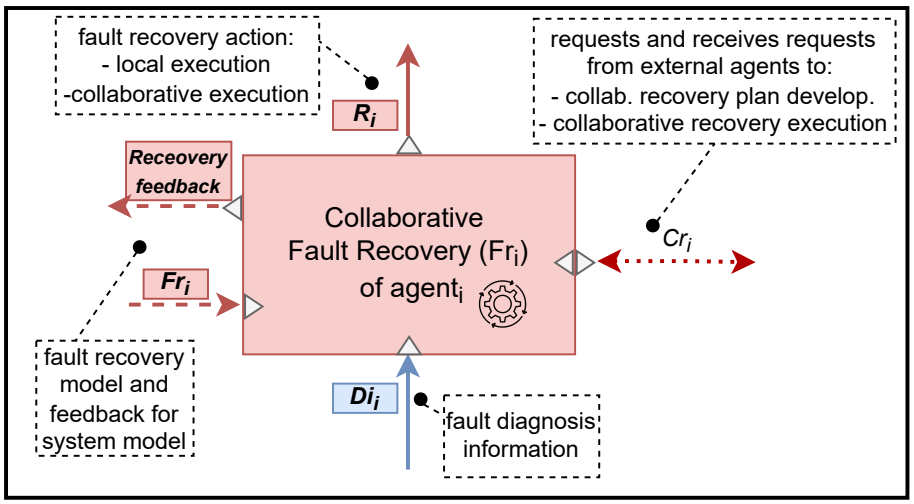


Figure 23 and depicted in Figure 27, the recovery output (R_i) for $agent_i$ is generated when the fault recovery model (Fr_i) determines an action that allows the agent to maintain its operation despite the occurrence of fault f_x, based on the diagnosis (Di_i) provided by the fault diagnosis stage. Consequently, the recovery output (R_i) can be expressed as a function of the fault recovery model (Fr_i), as outlined in Equation (3):

$$R_i = Fr_i(Di_i, CR_i(\tau,\sigma,\rho)) \rightarrow \begin{cases} re_i, \text{if local recovery} \\ re_{ji}, \text{if follow recomendation recovery action from } agent_j \\ \mathcal{RE}_i, \text{if is a recovery coordinator} \end{cases} \tag{3}$$

where the fault recovery model (Fr_i) domain is composed of:

- Di_i: the fault diagnosis output composed by $\{f_x,\mathcal{SI}_i,\mathcal{CA}_i,\mathcal{IM}_i\}$ as defined in the diagnosis stage.
- $CR_i(\tau,\sigma,\rho)$: denotes the collaboration model used when the $agent_i$ meet the collaboration criteria for recovery.

Based on the nature and impact of the detected and diagnosed fault, the recovery model Fr_i uses the domain information to propose and execute an appropriate recovery action plan, which can impact the component locally and other components that can contribute to the system's recovery. Therefore, the image of recovery model Fr_i consists of:

- re_i: recovery action that $agent_i$ needs to take to mitigate or recover from a detected and diagnosed fault with local impact. This recovery action can include, for example, adjusting operating parameters or restarting the component operation.

- re_{ji}: recovery action that $agent_i$ (as follower) should perform, recommended by $agent_j$(as coordinator role since $agent_j$ was the first to diagnose the global fault).

- $\mathcal{RE}_i$: a set of recovery actions that $agent_i$, with the role of coordinator in the face of a global fault, elaborate and send to agents that can collaborate in executing fault recovery. $\mathcal{RE}_i = \{re_{i1}, re_{i2}, ..., re_{ij}\} \mid j \in \mathbb{N}^* = \{1,...,n\}$.

The fault f_x may have a local or global impact, as reported by the fault diagnosis stage in $\mathcal{IM}_i$. If the fault f_x affects $agent_i$ locally, a local recovery action re_i will be formulated, for example, adjusting operational parameters like the speed or force of an actuator. However, if $agent_i$ cannot determine a suitable recovery action by itself, the collaborative recovery process (CR_i) is triggered. In this case, the collaboration parameters τ, σ, and ρ are employed, following the principles presented for the detection and diagnosis stages.

Suppose the fault f_x has a global impact and was initially diagnosed by $agent_j$. In that case, $agent_i$ can collaborate in the recovery process by following the recovery action (re_{ji}) advised by $agent_j$, who assumes the coordinator role. This procedure ensures a more collaborative and efficient system-wide fault recovery, where $agent_j$ proposes and orchestrates the recovery actions across the appropriate agents, leveraging each agent's available resources and operational capabilities.

However, in case $agent_i$ is the first to diagnose the global fault, it will be responsible for devising and coordinating the recovery action plan. In this scenario, $agent_i$ will generate a set of actions, denoted as $\mathcal{RE}_i$, which may involve agents within the set $\mathcal{A}$ to carry out collaborative actions to fault recovery. If $agent_i$ cannot formulate this set of recovery actions locally, it can initiate collaboration with other agents to elaborate the recovery actions. In this case, the collaboration process for recovery CR_i is activated, where the parameters τ, σ, and ρ are applied, as outlined in the collaboration procedure used in the detection and diagnosis stages. This condition ensures that the recovery process benefits from distributed and collaborative capabilities even in complex fault situations.

Then, to elucidate and rectify the idea of collaboration in the recovery stage, the collaboration Cr_i in the fault recovery module enables collaboration for distinct scenarios, which

can be categorized into two cases based on whether the diagnosed fault is local or global, as follows:

- **local fault - collaborative action plan development**: When the $agent_i$ faces a local fault and cannot independently formulate the recovery action plan, it initiates collaboration with external agents to develop a viable recovery strategy. For instance, a mobile robot diagnoses a local fault in its navigation system, preventing it from determining safe routes. Unable to develop a recovery plan independently, it requests help from nearby robots. The agents collaborate by sharing maps or navigation algorithms, enabling the faulty robot to recalculate its route.

- **local fault - request support to recovery**: In cases where $agent_i$ formulates a recovery plan but lacks sufficient resources or capabilities to execute it alone, the recovery plan involves external agents to collaboratively execute the required recovery actions as $\mathcal{RE}_i$. For example, a robotic arm diagnoses a local fault in its main actuator, preventing it from completing an assembly task. Although it can develop a recovery plan involving the temporary replacement of the actuator, it lacks the resources to execute it. In this case, it requests assistance from other agents in the system, such as another robot that can provide spare parts or act as a substitute.

- **global fault - collaborative action plan development**: This scenario resembles the case of "local fault – collaborative action plan development, with the key distinction being that the fault has a global impact, and the $agent_i$ as the first to diagnose the fault, assumes the role of coordinator.

- **global fault - coordinator of collaborative recovery**: If the $agent_i$ is the first to diagnose a global fault, it assumes the coordinator role for the recovery process. In this role, the $agent_i$ orchestrates the collaboration, recommends action, and requests support from other agents (expressed in $\mathcal{RE}_i$) to address the fault collectively. For instance, the $agent_i$ representing the monitoring temperature of a factory is the first to detect a fault in the cooling system of a factory (a global fault). The $agent_i$ assumes the coordinator role and requests collaboration from multiple agents, such as ventilation controllers and energy monitoring devices, to redistribute the thermal load until the fault is resolved.

- **global fault - participant (as follower) of collaborative recovery**: When another agent assumes the role of coordinator for addressing a global fault, and the $agent_i$ actively

participates in the collaborative recovery effort as a contributing agent following the instructions of a coordinator agent. For example, During a blackout in a distributed electrical system (a global fault), a substation ($agent_i$) receives a collaboration request from another substation (external agent) coordinating the recovery. The $agent_i$ collaborates by redirecting the energy flow within its area of responsibility, helping to restore the system's operation more quickly.

Table 6 summarizes the main functional features of the fault recovery stage.

Table 8 – Collaborative fault recovery stage features.

Feature	**Description**
Role	develop and execute the fault recovery action plan
Local input	diagnosis information ǀ collaboration model ǀ recovery model
Local output	action plan to execute to recovery from the fault ǀ feedback to system model
Local capabilities	fault recovery
External input	receive a request to collaborate in the fault recovery of an external agent ǀ receives action plan for recovery from a global fault (if follower)
External output	responds to the collaboration request for recovery ǀ send the recovery plan for other agents execute (if coordinator)
Collab. capabilities	request collaboration to improve local capabilities ǀ responds to collaboration requests to assist external agents in fault recovery ǀ collaboration for recovery from a global fault

Each module in the fault tolerance layer plays a crucial role in the FT operation of the agents in the CPS, enabling a structured and practical approach to dealing with disturbance episodes and faults, as established in the CF* meta-model.

3.5 AGENT INTERACTION FOR FT BASED ON CF*

In the MAS-based CPS adopting the CF* approach for distributed FT, each agent operates autonomously and possesses its own fault detection, diagnosis, and recovery models, trained on datasets that may be unique to each agent. Additionally, each agent has only partial knowledge of the overall system. When an agent cannot locally execute one of the FT stages, it interacts with other agents in the system to enhance its individual capabilities.

Collaboration, a concept rooted in social and natural systems, also extends to CPSs. The agents must coexist in an environment where they can aggregate information to fulfill their tasks. In the context of FT, the agents consolidate their partial knowledge through interaction to address faults collaboratively. The collaboration process occurs when two or more agents with partial knowledge about the global system or insufficient resources to achieve their individual

goals interact, where an agent can require or provide information and support through interaction with other agents. Collaboration is characterized by an agent assisting another agent in executing specific tasks without expecting to have individual benefits. Additionally, in CF*, the coordination concept can manifest with the need to synchronize a sequence of actions to be performed by the agents, for example, in the fault recovery process.

The CF* approach supports agent interaction through asynchronous communication mechanisms, meaning that an agent sending a message over a communication infrastructure can continue its task execution. Communication can occur via point-to-point, broadcast, or multi-cast connections.

Interaction relies on standardized messaging facilitated by an appropriate ACL, which defines a structured framework for message organization and leverages performatives to specify the intent and context of communication. These interaction protocols can derive their structure from established agent communication protocols, particularly for this work, like those delineated by FIPA.

The following Sections describe different interaction protocols that can support the collaboration between the agents for CF*. The interactions are classified into essential interactions (that enable the sharing of agent information) and those specific to fault tolerance stages: interaction for fault detection, interaction for fault diagnosis, and interaction for fault recovery. This work models these interactions using Agent Unified Modeling Language (AUML) interaction diagrams (ODELL *et al.*, 2000), an extension of Unified Modeling Language (UML) sequence diagrams tailored for multi-agent systems.

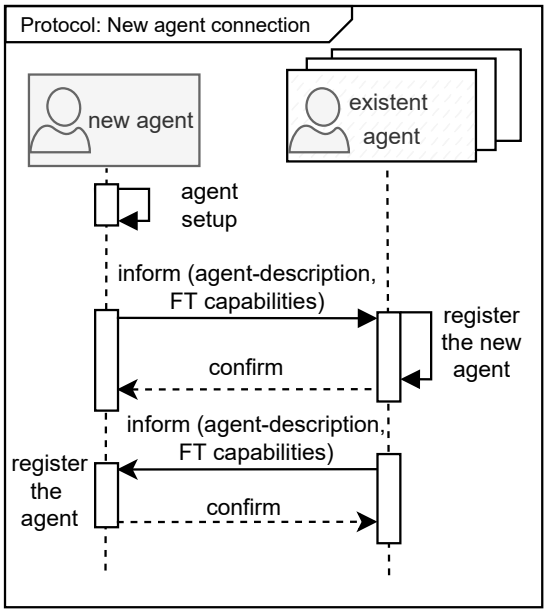
3.5.1 Interaction for Agent Information

Each agent possesses models and algorithms for detecting, diagnosing, and recovering from faults locally and collaboratively. To ensure that all agents in the system are informed of the capabilities of the available agents, it is necessary to disseminate information during both the launching and disconnection processes of each agent. This synchronization step is crucial for maintaining the consistency and functionality of the distributed system.

When an agent is launched, i.e., when a new agent with CF* connects to the CPS, it must inform its capabilities so that other agents can register its functionalities. As illustrated in Figure 28, the new agent broadcasts its capabilities to all existing agents in the CPS, providing details about its fault tolerance mechanisms, such as detection, diagnosis, and recovery algorithms,

as well as the data structures used for inference. This enables other agents to consider its compatibility and potentially request its collaboration in the future. Additionally, once the new agent completes this broadcast, it receives similar information from the existing agents in the system. This allows the new agent to create a repository or directory of functionalities "yellow pages" and capabilities, enabling the new agent to identify compatible agents for future collaboration.

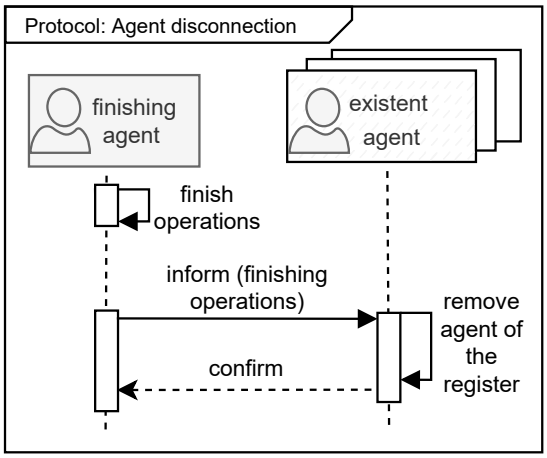
Figure 28 – Protocol for sharing information of new agent launched.



When an agent concludes its operation or is finished within the CPS, it must notify the system of its disconnection. As shown in Figure 29, the remaining agents in the system will update their registries to remove the disconnected agent. This ensures that future collaboration processes do not attempt to select an agent that is no longer active, thereby preventing potential errors or inefficiencies in the CPS. By formally disconnecting, the integrity and consistency of the distributed system are maintained, allowing the active agents to continue operating smoothly without mistakes caused by outdated information.

These agent interaction procedures of connections and disconnection are essential to enable the dynamic plug-and-play functionality within the CF* and serve as the foundation for collaborative interactions for fault tolerance. By allowing agents to seamlessly join and leave the system while ensuring that their FT capabilities are transmitted and synchronized with other agents, the MAS-based CPS can maintain its adaptability, flexibility, and scalability where the availability of agents may change over time.

Figure 29 – Agent Disconnection protocol.



3.5.2 Interaction for Collaborative Fault Detection

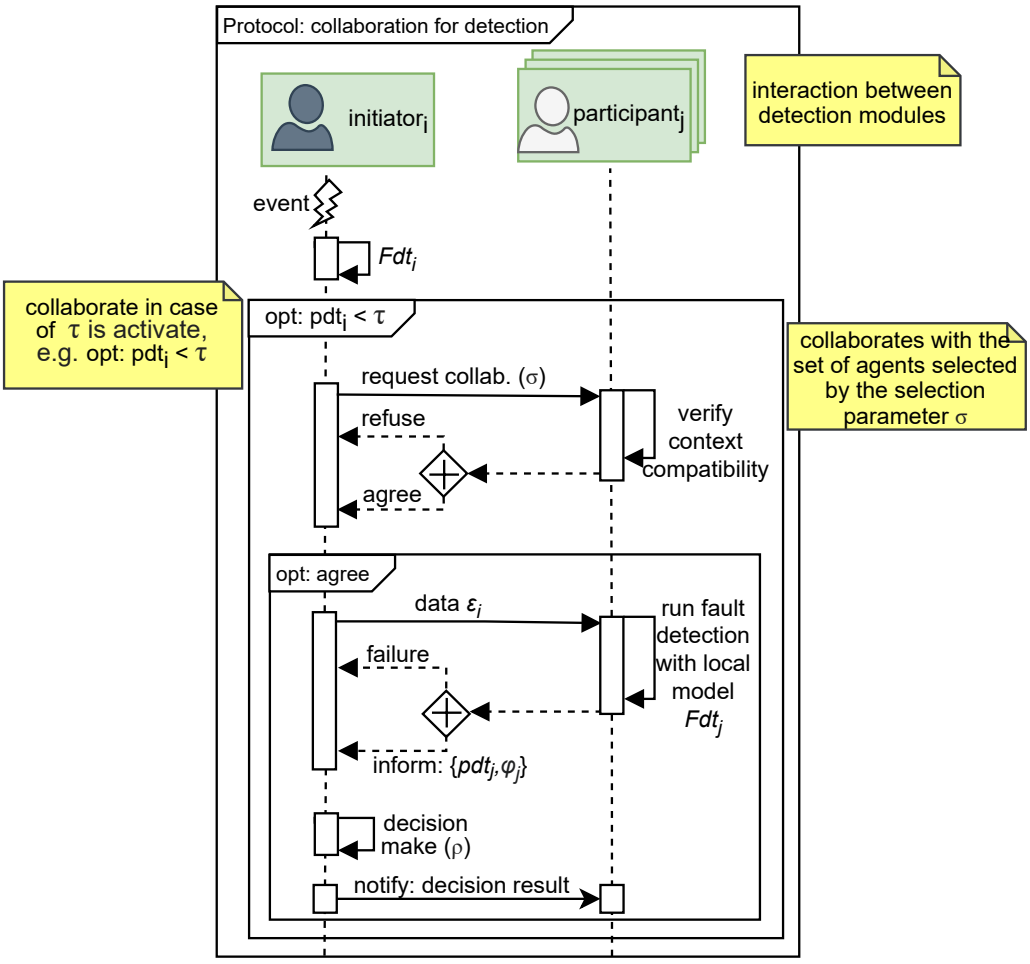
In addition to the standard FIPA-ACL interaction protocols (e.g., FIPA Request to interact with the selected agents to confirm whether these agents also observed any disturbance for complementing the fault detection decision), the CF* defines a collaboration protocol to enhance the fault detection, as illustrated in Figure 30. This proposed interaction aims to improve the individual fault detection capability by enabling the initiator agent to interact with a set of participating agents, each equipped with compatible detection models. These models may employ different algorithms or be trained on distinct datasets, providing complementary perspectives and enhancing the fault detection accuracy of the initiator agent.

After executing detection model Fdt_i and activating collaboration trigger parameter τ[5], the initiator agent requests collaboration for the selected agents (according to the collaborative parameter σ), i.e., the participants. Considering that the participant fault detection model is compatible with the initiator model, the initiator sends the observation raw data (ϵ_i) for the participants to infer.

Since the participants' fault detection models may have been trained with different datasets, parameters or even with different detection algorithms, this interaction provides a valuable complement for decision-making. After each participant responds to the initiator, the decision is made according to the collaborative parameter decision-make (ρ). The outcome of the decision-making process is communicated to all participating agents to ensure transparency and collective awareness within the MAS. It also provides valuable data for the system model, updating each agent's detection model and collaboration parameters, thereby improving fault

[5] Consult Section: 3.2.2 Collaborative Parameters for the CF*

Figure 30 – Collaboration fault detection protocol.



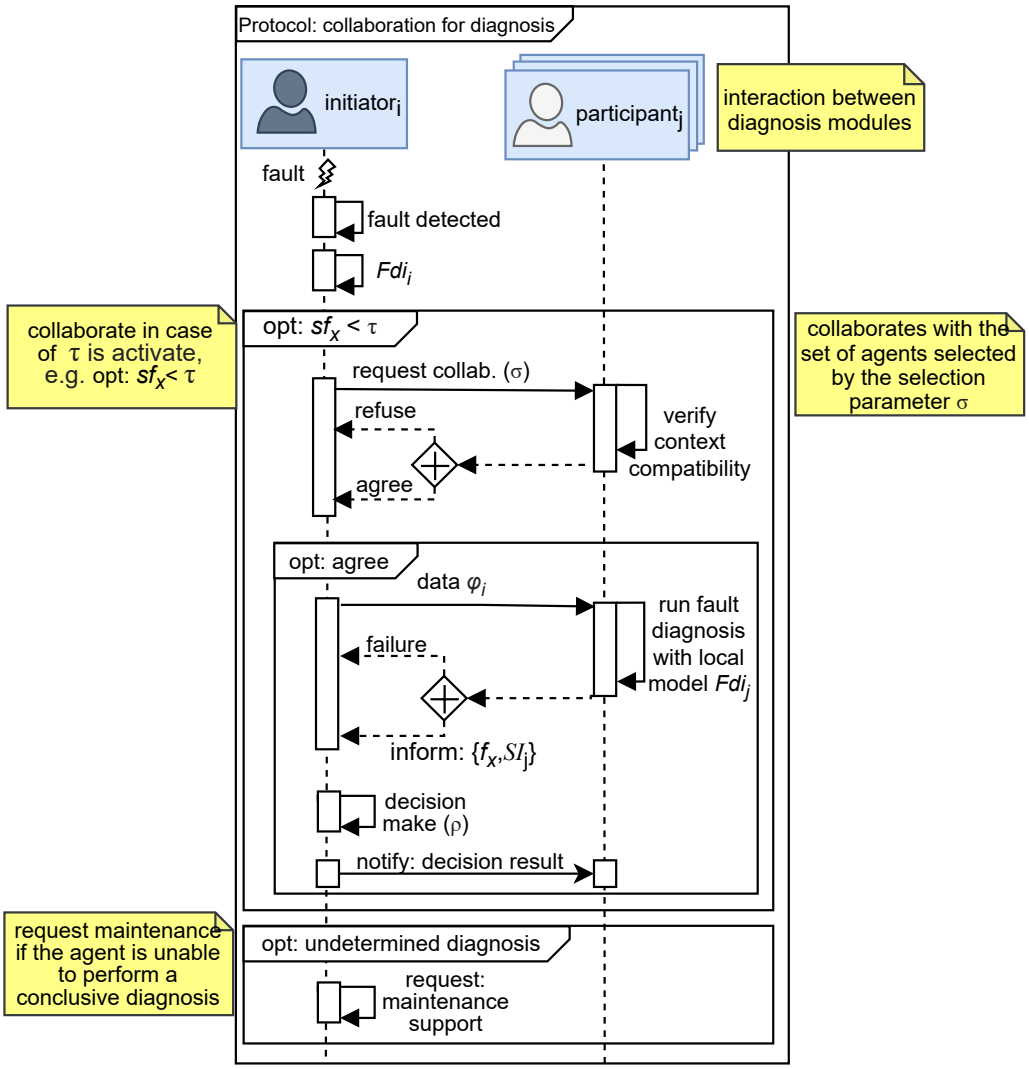
detection capabilities over time. The possibility of adopting collaboration between the agents (with different algorithms, training parameters, datasets, and experiences) makes it possible for the detection models to adapt to address the challenge of detecting faults.

3.5.3 Interaction for Collaborative Fault Diagnosis

The CF* introduces a customized collaboration protocol specifically designed for fault diagnosis, shown in Figure 31. This protocol enables different agents within the system to communicate and exchange information pertinent to diagnosing issues.

After executing the diagnosis model Fdi_i and activating the collaboration trigger τ, the initiator agent requests collaboration with compatible agents. The initiator agent shares data associated with the fault (φ_i) with all participant agents selected according to the σ parameter. Once responses are received from the participants, the initiator agent makes the diagnosis decision using the ρ strategy. This decision is then shared with all participant agents, promoting

Figure 31 – Collaboration fault diagnosis protocol.



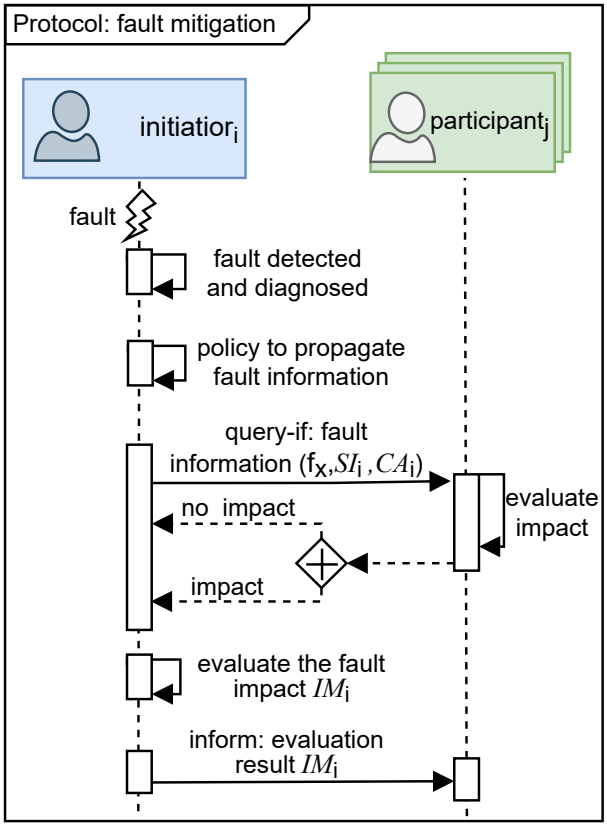
transparency and collective awareness within the MAS and providing useful information for updating each agent's diagnosis models and collaboration parameters, thereby improving fault diagnosis capabilities over time.

In alignment with each agent's autonomy, the initiator agent is responsible for determining the diagnosis by applying decision-making (ρ) based on the inputs gathered from participating agents during collaboration. Different agents may have varied experiences and diagnosis models, and conflicting diagnoses may arise. When the decision policy is insufficient to resolve these conflicts and ensure diagnosis consistency or collaboration is not achieved, whether due to refusal from all participant agents, the initiator agent will escalate the issue to external maintenance support to analyze the event and provide an appropriate diagnosis.

Additionally, CF* establishes a protocol to determine whether a fault diagnosed is local or global, which is critical for appropriate recovery actions. After diagnosing a fault, the

initiator agent (the one identifying the fault) assesses the potential for the fault to impact other agents in the system, as illustrated in Figure 32. The initiator shares fault diagnosis information $(f_x, \mathcal{SI}_i, \mathcal{CA}_i)$ with the participant agents (i.e., potential agents whom the failure might impact), who independently evaluate whether the fault has an impact on their own operation. Each participant then reports back to the initiator with their assessment. If these responses confirm fault propagation across multiple agents, the initiator classifies the fault as global rather than local. This classification is essential for the subsequent fault recovery stage, as it indicates that collaborative actions may be required to avoid the fault's broader effects and protect the system's overall functionality.

Figure 32 – Protocol to mitigate the fault impact.

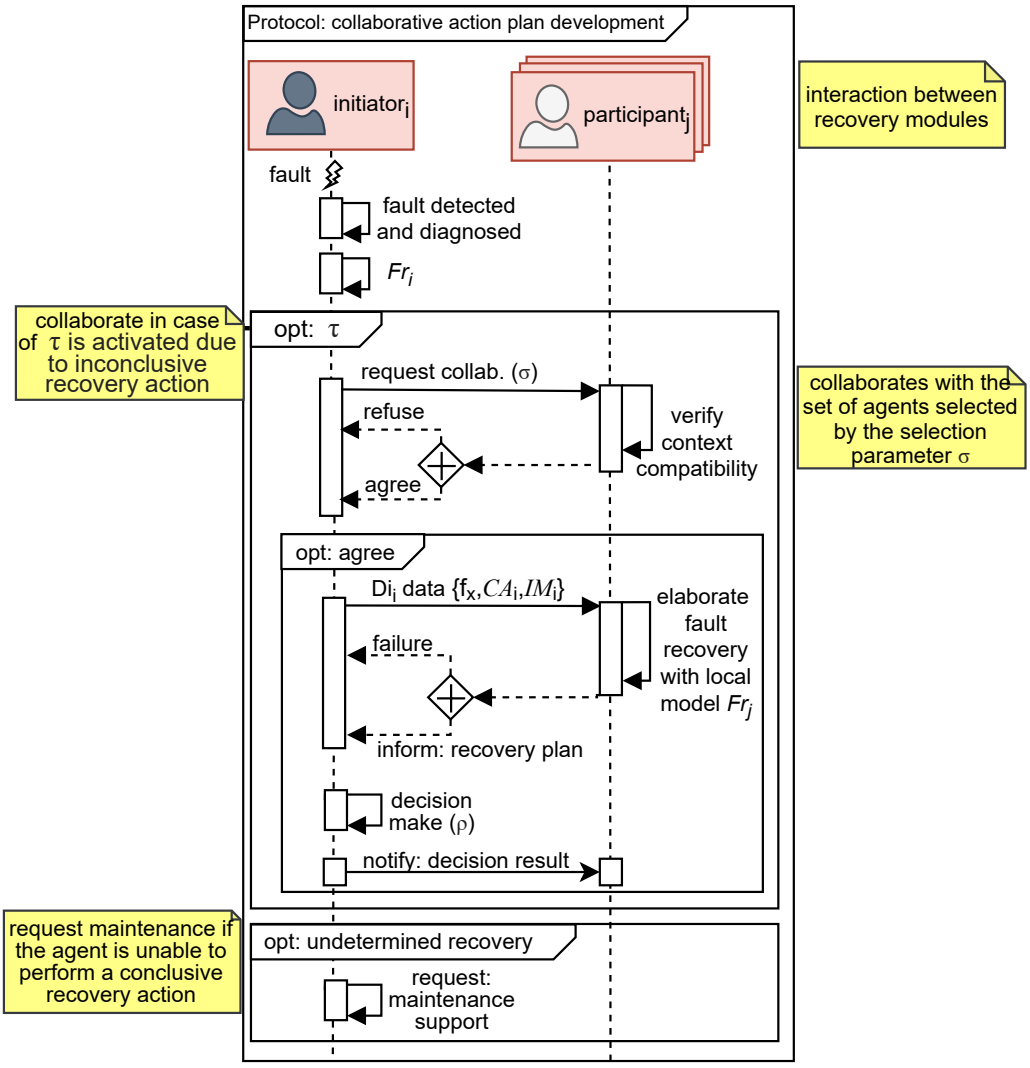


3.5.4 Interaction for Collaborative Fault Recovery

The CF* introduces two customized collaborative protocols specifically designed to support fault recovery processes. The first protocol, illustrated in Figure 33, addresses collaboration for elaborating a recovery plan, whether the plan is to be executed locally or involves multiple agents. The second protocol, presented in Figure 34, focuses on collaboration during

the execution of the recovery plan, which is necessary when agents are required to address either a local fault or a fault with broader impacts.

Figure 33 – Protocol for collaborative recovery action plan elaboration.

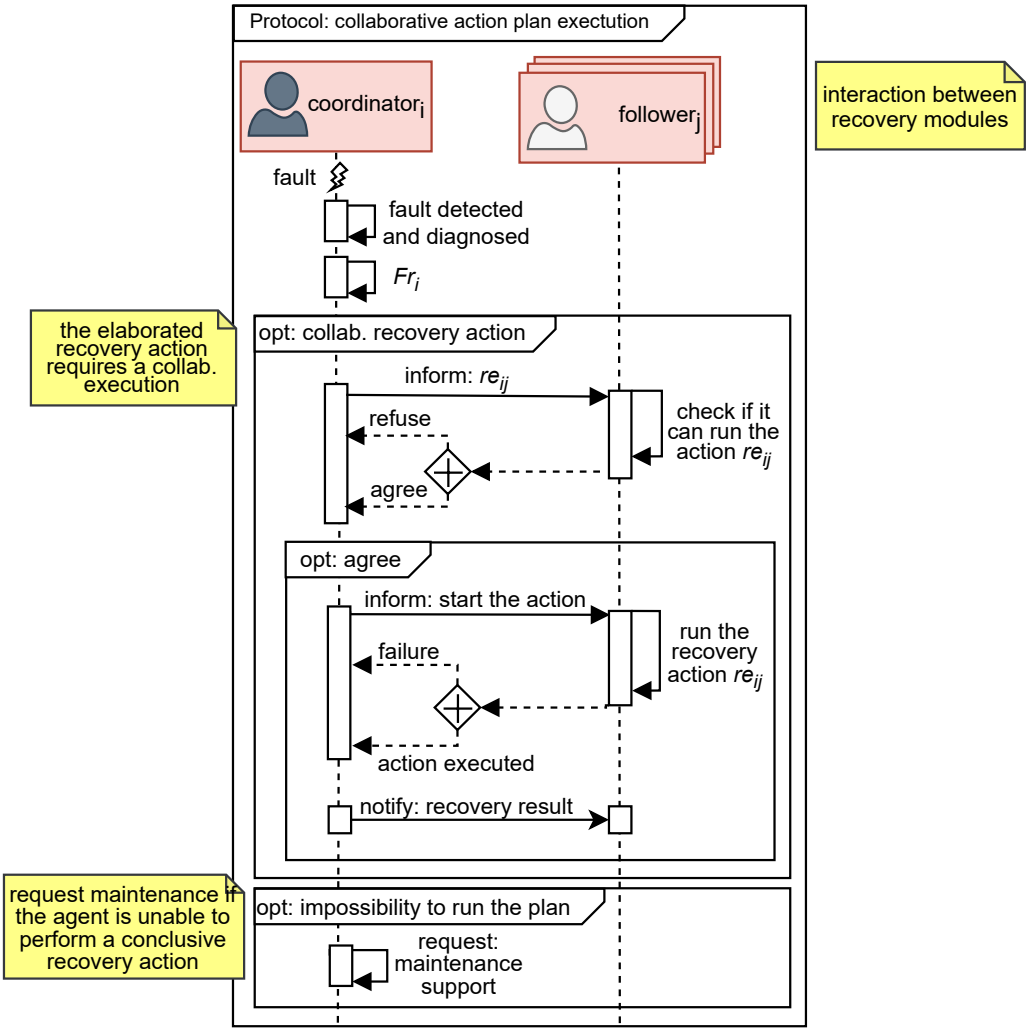


In the protocol for collaborative recovery action plan elaboration (Figure 33), the initiator agent has already detected and diagnosed the fault and must now establish measures to mitigate its impact. If the agent cannot formulate a plan locally and then the collaboration trigger parameter (τ) is activated, such as a timeout or low confidence in the locally generated plan, the initiator agent will proceed to select participant agents based on the selection parameter (σ) to assist in the plan's elaboration. Once selected, the participant agents confirm their capacity and availability to collaborate, after which the initiator shares the fault diagnosis information ($f_x, \mathcal{CA}_i, \mathcal{IM}_i$). Each participant then generates a proposed recovery plan using their individual models and sends it back to the initiator. The initiator consolidates the responses and, guided by the decision parameter (ρ), defines the final recovery plan. If no viable plan can be formulated,

the initiator escalates the issue by requesting external maintenance support.

The second protocol of the fault recovery stage, depicted in Figure 34, governs collaborative recovery execution, where a coordinating agent (coordinator) requires the assistance of other agents (followers) to execute the recovery plan. To assume the role of coordinator, an agent must be the first to diagnose a fault with broader system-wide impacts (i.e., global fault) or be unable to independently recover from a local fault.

Figure 34 – Protocol for collaborative recovery execution.



After detecting, diagnosing, and formulating the recovery action plan, the coordinator communicates the required actions (re_{ij}) to the selected follower agents. The followers confirm their commitment and ability to participate, then the coordinator specifies the timing for executing the assigned tasks. During the execution, followers report any issues encountered or confirm task completion to the coordinator. Once all recovery actions are completed, the coordinator notifies all participants that the process is concluded. However, if a recovery action cannot be successfully executed, the coordinator escalates the situation by requesting external support.

The execution of the collaborative recovery action plan involves increased complexity and sophisticated interaction, demanding seamless coordination, compatibility, and efficient communication between the agents evolved.

3.6 EVALUATION METRICS FOR COLLABORATION

The collaboration enables the collective intelligence of the agent to operate, allowing each agent, through the information exchange, to transcend its individual capacity in fault tolerance. Similarly, when established incorrectly, collaboration may not bring benefits or even negatively impact; for example, excessive interactions can consume significant communication bandwidth and processing resources or even decrease task performance by inducing errors in detection or diagnosis stages. Therefore, it is essential to establish metrics to evaluate the behavior and outcomes of the collaboration among agents in CPS.

In the context of the CF* approach, where fault detection and diagnosis can be acheived by collaboration across multiple agents, understanding the effectiveness of collaboration becomes key point. Since each agent may have different models, levels of confidence, or perspectives over the same event, the collaboration process needs to be evaluated not only in terms of accuracy but also in terms of consistency and collaboration impact.

Then, this thesis introduces the 7CM as a novel metric system designed specifically for the CF* approach. These metrics allow agents to quantitatively assess the impact of collaboration on their own inference process, identify when collaboration was beneficial or detrimental, and enable self-adjustments of their collaborative strategies. The 7CM is, therefore, a key element of this thesis, aligning with its core objective of enabling selective and contextual collaboration among agents for FT.

The proposed method to evaluate the collaboration is specific for the stages of Fault Detection and Diagnosis (FDD) of CF*. The 7CM comprises seven metrics that allow assessing and adjusting the collaborative behavior in each agent's detection or diagnosis models. These metrics are based on the ratio between the quantity of inferences (qi) and the collaboration outputs regarding the quantity of collaboration (qc), i.e., the number of times the agent's collaboration trigger (τ) is fired. The 7CM analyzes whether the collaborations remain or modify the result initially classified by the agent detection or diagnosis local model to verify the benefit of the collaboration process for the agent.

Figure 35 presents the algorithm designed to compute the 7CM, which is applied

independently to either the fault detection or fault diagnosis stages. Initially, the algorithm retrieves all the inferences (related to detection or diagnosis) performed by $agent_i$ (line 3 of Figure 35). The quantity of inferences made by $agent_i$ is then assigned to the variable qi. From lines 5 to 24, the algorithm iterates through the $inferences$ list, analyzing each $inference$ to determine whether collaboration was required for FDD. If the collaboration was necessary, the algorithm assesses its impact by verifying whether the collaboration resulted in remaining or modifying the initial classification performed locally by $agent_i$. This evaluation includes determining if the collaboration outcome remained correct, incorrect, or modified correctly or incorrectly compared to the local classification of the $agent_i$. Finally, between lines 25 and 31, the algorithm calculates the 7CM quantifying the collaboration's role in the inference process. These metrics provide a comprehensive assessment of the collaborative interactions, including the frequency of collaboration, correctness, and influence on the $agent_i$ decision-making.

Figure 35 – Algorithm of the 7CM to evaluate the collaboration for FDD.

```
1: procedure 7CM(agent_i):
2:     qi, qc, ri, rc, mi, mc ← 0;                          ▷ Init the 7CM variables
3:     inferences ← GetListInferences(agent_i);
4:     qi ← len(inferences);                                 ▷ Quantity of inferences
5:     for inference in inferences do:
6:         localResult ← GetLocalResult(inference);
7:         if RequiredCollaboration(localResult) then:
8:             qc ← qc + 1;                                  ▷ Increment the quantity of collab.
9:             CollabResult ← GetCollabResult(inference);
10:            if WasIncorrect(localResult) then:
11:                if WasIncorrect(CollabResult) then:
12:                    ri ← ri + 1;                          ▷ Collab. remained the result incorrect
13:                else if WasCorrect(CollabResult) then:
14:                    mc ← mc + 1;                          ▷ Collab. modified the result correct
15:                end if
16:            else if WasCorrect(localResult) then:
17:                if WasIncorrect(CollabResult) then:
18:                    mi ← mi + 1;                          ▷ Collab. modified the result incorrect
19:                else if WasCorrect(CollabResult) then:
20:                    rc ← rc + 1;                          ▷ Collab. remained the result incorrect
21:                end if
22:            end if
23:        end if
24:    end for
25:    CM1 ← qc/qi;                                          ▷ Calculate the 7CM metrics
26:    CM2 ← ri/qc;
27:    CM3 ← rc/qc;
28:    CM4 ← mi/qc;
29:    CM5 ← mc/qc;
30:    CM6 ← (ri + mi)/qc;
31:    CM7 ← (rc + mc)/qc;
32:    return CM1, CM2, CM3, CM4, CM5, CM6, CM7
33: end procedure
```

The first metric, CM_1, evaluates the agent's anxiety, checking if the collaboration triggers τ is activated excessively. The reason for the quantity of collaboration (qc) is analyzed, i.e., the inferences that activated the collaboration trigger with the quantity of inferences (qi), as presented by Equation (4). If this ratio is high, it indicates that the agent presents excessive collaboration, requiring an adjustment to the τ parameter, or that the detection or diagnosis model needs to be updated to become more confident and not trigger the collaboration so often. On the other hand, if this ratio is low, the agent is not requesting collaboration and, therefore, contains only local behavior. $\mathcal{D}(CM_1)$ defines the domains of the variables.

$$CM_1 = \frac{qc}{qi} \rightarrow [0,1] \mid \mathcal{D}(CM_1) = \begin{cases} \{qi \in \mathbb{N}^*\} \\ \{qc \in \mathbb{N}^* \mid 0 < qc \leq qi\} \end{cases} \tag{4}$$

The CM_2 metric represents the collective error and checks whether the agent selection and decision-making in collaboration could be more efficient. It is analyzed when the local model is incorrect, and the collaboration maintains the incorrect fault classification decision. The Equation (5) shows whether, after the collaboration, the decision remained incorrect (ri). In this case, τ is performing as expected, however, the selection σ and decision-making parameter ρ are inefficient and must be updated.

$$CM_2 = \frac{ri}{qc} \rightarrow [0,1] \mid \mathcal{D}(CM_2) = \begin{cases} \{ri \in \mathbb{N} \mid \ ri \leq qc\} \\ \{qc \in \mathbb{N}^* \mid 0 < qc \leq qi\} \end{cases} \tag{5}$$

Equation (6) presents the CM_3 metrics that indicate when the collaborative behavior of the agent tends to confirm if the local classification decision is correct since it analyzes whether the decision after the collaboration remained correct (rc). The high ratio may indicate an insecure agent, meaning the τ collaboration trigger must be adjusted.

$$CM_3 = \frac{rc}{qc} \rightarrow [0,1] \mid \mathcal{D}(CM_3) = \begin{cases} \{rc \in \mathbb{N} \mid \ rc \leq qc\} \\ \{qc \in \mathbb{N}^* \mid 0 < qc \leq qi\} \end{cases} \tag{6}$$

The collaboration between agents can modify incorrectly (mi) the decision, leading the agent that starts the collaboration into error. Equation (7) presents the CM_4 metric and defines it as the worst collaboration scenario when this value is high. To correct this, updating the trigger τ or decision-making ρ is necessary since the model is already classifying correctly without collaboration.

$$CM_4 = \frac{mi}{qc} \rightarrow [0,1] \mid \mathcal{D}(CM_4) = \begin{cases} \{mi \in \mathbb{N} \mid \ mi \leq qc\} \\ \{qc \in \mathbb{N}^* \mid 0 < qc \leq qi\} \end{cases} \tag{7}$$

The CM_5 metric presented in Equation (8) points out the most desired behavior of the collaboration, as the decision of the collaboration modifies correctly (mc) the local classification. This metric highlights that the collaboration prevents a fault from going undetected or from false fault alerts or even incorrect diagnoses. In this sense, when this ratio is high, it indicates that τ, σ, and ρ are set efficiently. The result of the collaboration can be used to retrain and improve the detection model.

$$CM_5 = \frac{mc}{qc} \rightarrow [0,1] \mid \mathcal{D}(CM_5) = \begin{cases} \{mc \in \mathbb{N} \mid \ mc \leq qc\} \\ \{qc \in \mathbb{N}^* \mid 0 < qc \leq qi\} \end{cases} \tag{8}$$

The CM_6 metric adds the undesired collaboration decision, that is, ri and mi. At the same time, the CM_7 metric adds the desired decision, that is, rc and mc. Therefore, an agent that can get a low CM_6 and a high CM_7 has good detection and diagnosis models and collaboration parameters τ, σ, and ρ well configured, as delineated by Equation (9) and Equation (10) respectively.

$$CM_6 = \frac{ri + mi}{qc} \rightarrow [0,1] \mid \mathcal{D}(CM_6) = \begin{cases} \{ri + mi \in \mathbb{N} \mid \ ri + mi \leq qc\} \\ \{qc \in \mathbb{N}^* \mid 0 < qc \leq qi\} \end{cases} \tag{9}$$

$$CM_7 = \frac{rc + mc}{qc} \rightarrow [0,1] \mid \mathcal{D}(CM_7) = \begin{cases} \{rc + mc \in \mathbb{N} \mid \ rc + mc \leq qc\} \\ \{qc \in \mathbb{N}^* \mid 0 < qc \leq qi\} \end{cases} \tag{10}$$

Table 9 summarizes the 7CM approach, presenting the behavior indicated by each metric, describing each one, and highlighting the parameters that have the most significant impact on their influence.

The proposed metrics support agents' continuous improvement during their collaborative fault detection and diagnosis operation. The 7CM can assist each agent in adapting during its execution, identifying moments when more collaboration is needed or when this behavior needs to be reduced. Additionally, it can provide indications of when the FDD models must be updated.

Table 9 – Summary of the 7CM to evaluate collaborative fault detection and diagnosis in CPS.

Metrics	***Behavior represented***	***Description***	***Parameters with the most influence***
CM_1	anxiety	evaluate the collaboration rate	τ FDD model
CM_2	collective error	locally wrong and after the collab. the error remains	σ ρ
CM_3	insecurity	locally correct, and after the collab. confirm the correct	τ FDD model
CM_4	error induction	locally correct, but when collab. it is induced to error	τ σ ρ
CM_5	correction	locally wrong, and when collab. it is corrected	τ σ ρ
CM_6	collab. undesired	collab. that negatively impact the agent	all parameters
CM_7	collab. desired	collab. that positively impact the agent	all parameters

3.7 SUMMARY

In this chapter, the Collaborative Fault-Star (CF*) approach for fault tolerance in CPSs was introduced. This approach focuses on establishing collaboration during the fault detection, diagnosis, and recovery stages, thereby promoting the system's flexibility, scalability, robustness, and pluggability, to improve the precision and accuracy of FT mechanisms within the CPS.

This Chapter presented the MAS-based CPS architecture designed to support the CF* approach. This included defining the architecture's computational and physical layers, detailing the elements that constitute each layer, and contextualizing the role of an agent within this architecture. The concepts of cyber and physical components were also defined, along with their interaction to constitute the cyber-physical components. Subsequently, the Chapter described how the cyber components interact and introduced the Edge-Cloud continuum infrastructure for deploying agents in the presented CPS architecture.

Furthermore, this Chapter presented the generic meta-model of the CF* approach, which outlines the distributed FT mechanism and the key modules enabling agents' autonomy, digitization, control, intelligence, and collaboration within the CPS architecture. The proposed meta-model supports the generalization and aggregation of concepts, allowing the approach to be applied and instantiated across diverse CPS contexts and applications.

The CF* establishes that each agent embodies autonomous, intelligent, and collaborative

behavior, with local fault tolerance capabilities structured into three stages: detection, diagnosis, and recovery. When a disturbance occurs, these stages are sequentially activated, beginning with fault detection, followed by diagnosis, and concluding with fault recovery. This conceptual model of CF* incorporates the key requirements and features addressing the ***RQ-1*** .This Chapter provided a detailed description of each fault tolerance stage within the CF*, complemented by the formal modeling. The $SystemModel$ module of CF* is responsible for continuously updating and adapting the models for each FT stage. Each stage can collaborate with corresponding stages of external agents, based on the collaboration parameters, namely, collaboration trigger (τ), agents selection (σ), and decision-make (ρ) to improve its performance or accuracy. These collaborative interaction protocols described in this Chapter enable the agents to transcend their individual capabilities, enhancing performance when facing uncertainty or predefined collaboration in detection, diagnosis, or recovery. This Chapter also introduced customized collaboration protocols tailored to each fault tolerance stage.

Finally, the Seven Collaboration Metrics (7CM) metrics were presented as a means to evaluate whether collaboration during the detection and diagnosis stages translates into benefits for the agents. The metrics also provide insight into how the collaborative behavior can be adjusted or how the inference models can be updated to improve the robustness of each agent.

In the next Chapter, the dynamic behavior of the agent with CF* will be modeled using the Petri Net formalism. Validation of the developed models will also be performed to verify their correctness.

4 AGENT MODEL AND DYNAMIC BEHAVIOR SPECIFICATION FOR CF*

The CF* approach is founded on a network of autonomous, intelligent, and collaborative agents that represent the components of a CPS. Each agent is endowed with self-FT capabilities, enabling it to address faults individually while collaborating to enhance its overall FT capabilities. In the previous Chapter, the principles and foundation of the CF* approach were introduced, detailing all components of the proposed approach. This included the MAS-based CPS architecture and the individual and collaborative agent behaviors necessary to support FT, specifically addressing the FT stages, collaboration parameters, communication protocols, and performance metrics designed to refine detection, diagnosis, and recovery models.

However, CPS applications deployed using agents are widely recognized as inherently complex systems. This complexity arises from concurrent and asynchronous tasks and operations, making these systems challenging to design, implement, and understand (LEITÃO, 2004). As such, a formal modeling of the agent's structural and behavioral aspects is needed. It provides a means to clarify and comprehensively describe the CF* functionalities, thereby serving as a cornerstone for the effective design and validation of CPS applications.

This Chapter presents a formal model for specifying and analyzing the functionalities of a generic agent operating under the CF* approach. Particular attention is devoted to examining the behavior of each collaborative FT stage using the Petri nets (PN) formalism. Adopting the PN formalism enables a rigorous specification and validation process by leveraging its robust mathematical foundation and graphical representation. The PN ensures accurate verification of the correctness of the proposed agent model and the specifications governing the behavior of individual agents within the CPS architecture, analyzing the agent model's structural, behavioral, and performance properties.

The remainder of this Chapter is structured as follows:

- Section 4.1 introduces the key concepts of PN modeling that are essential for understanding the content presented in this Chapter.
- Section 4.2 presents the proposed PN model, which encapsulates the fundamental behavior of a generic agent operating with the CF* approach.
- Section 4.3 provides a refinement of the proposed PN, focusing on the collaborative fault detection, diagnosis, and recovery stages of CF*.

- Section 4.4 conducts a comprehensive analysis of the structural, behavioral, and performance properties of the proposed PN, ensuring its correctness and consistency.
- Section 4.5 summarizes the main aspects and findings discussed in this Chapter.

4.1 PETRI NETS TO MODEL THE AGENT DYNAMIC BEHAVIOR

A model constitutes an abstract representation of a segment of reality designed to facilitate its comprehension. The primary benefit of modeling reality or a system lies in its capacity to render the concept more explicit, simplified, and suitable to manipulation, as opposed to the complexity inherent in the actual reality it aims to represent (FERBER; WEISS, 1999).

The modeling of the dynamic behavior of the agent, a formal modeling tool, is essential, encapsulating features such as concurrency, asynchronous behavior, conflicts, deadlocks, and resource allocation (SILVA; VALETTE, 1988). Furthermore, this formalism should be equipped to validate these behavioral characteristics and evaluate critical factors, including deadlock identification and performance analysis.

Considering this, the UML (RUMBAUGH, 2005) is a suitable tool for modeling object-oriented systems; however, it does not effectively support modeling their dynamic behavior nor the formal validation of these specifications. Moreover, UML inadequately captures the autonomous and proactive behaviors of agents (WOOLDRIDGE; CIANCARINI, 2000). State machines, another tool extensively employed for modeling agents, encapsulate the concept of an event and, to a limited extent, the notion of activity, defined as a state between two events. Nevertheless, state machines are constrained in representing a process encompassing multiple parallel processes and concurrent evolution. In particular, a finite state machine primarily delineates a single sequential process (CARDOSO; VALETTE, 1997).

Conversely, PN serves as a comprehensive modeling tool, both graphical and mathematical, particularly suited for analyzing the behavior of autonomous agents that are concurrent, asynchronous, stochastic, and highly distributed, as found in MAS-based CPS. Unlike UML and state machines, PN offers the ability to design, validate, and verify the agent behavior, leveraging the robust mathematical principles inherent in its formalism. Hence, the methodology presented here employs a specific variant of PN, designed for system production control modeling as introduced in (COLOMBO; CARELLI, 1997) to effectively model agent behavior using the CF* approach.

Petri nets are directed graph defined by a four tuple, $PN = \{P, T, I, O\}$, where

- $P = p_1, ..., p_m$ is a finite set of places.
- $T = t_1, ..., t_n$ is a finite set of transitions.
- $I : (P \times T) \rightarrow \mathbb{N}$ is an input function that defines directed arcs from places to transitions. Each element of I represents the weight of the input arc from the place p_a to the transition t_b.
- $O : (P \times T) \rightarrow \mathbb{N}$ is an output function that defines directed arcs from transitions to places. Each element of O represents the weight of the output arc from the transition t_b to the place p_a.

In a graphical representation of a PN, places P are illustrated as circles, and transitions T are depicted as bars. The links connecting places to transitions, denoted as I and O, are represented using directed arcs, which are indicated by arrows. Direct arcs between two places or between two transitions are prohibited. Each arc carries an integer value representing its weight, and if unspecified, the default weight is assumed to be one (LEITÃO, 2004).

A marked PN is distinguished by the integration of tokens, represented as black dots, alongside the places, transitions, and arcs. These tokens are located within places, traverse through arcs, and their movement within the PN is regulated by transitions (CARDOSO; VALETTE, 1997).

The dynamic development of the PN is articulately described by the subsequent enabling and firing principles:

- A transition is triggered when each input place contains a quantity of tokens that is at least equal to the weight attributed to the corresponding arc.
- When a transition is activated, it initiates a process that subtracts a specific quantity of tokens from each associated input place. Simultaneously, an equivalent quantity of tokens, as dictated by the output arcs, is deposited into each linked output place.

The reachability set associated with an initial marking m_0 consists of all markings obtainable from m_0 and is symbolized as $R(m_0)$. A PN is defined as reversible if, for every m contained in $R(m_0)$, there is a sequence of transitions that can be executed to satisfy $m_0 \in R(m)$ (LEITÃO, 2004).

Within this study, tokens, places, and transitions are attributed to specific meanings and contexts crucial for accurately interpreting the agent model. Their roles are delineated as follows:

- Places represent the states of the agent. The existence of tokens in a place indicates the status of a place (active or not).
- A transition firing represents an agent activity or operation.

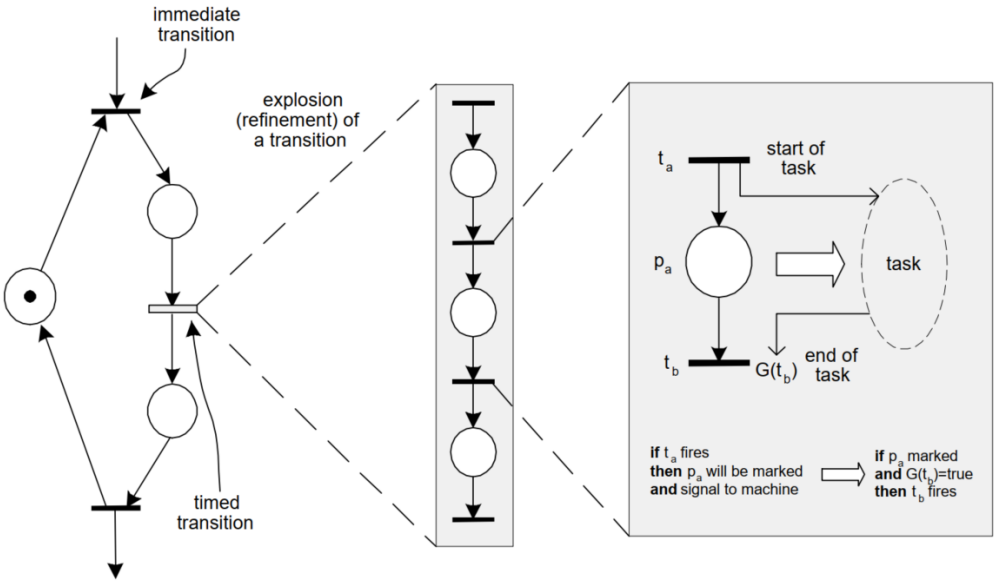
A modeled agent may consist of activities or operations, represented by transitions, with varying degrees of complexity, which occur much faster (or slower) than others. Furthermore, it may be necessary to introduce transitions corresponding to the purely logical dimensions of the agent's behavior, which are absent of temporal association, and transitions related to more complex operations that require execution times (COLOMBO *et al.*, 1997). Within these conditions, the PN utilized in this study caters to two distinct categories of transitions:

- **Immediate transitions** represented by a thin bar to model atomic activities, such as dispatching a message or initiating a procedure.
- **Timed transitions**, drawn by a thick bar, which have associated time that specifies the amount of time that must elapse before the transition fires. This type of transition is used to represent temporal activities, i.e. time consuming activities. As an example, execution of an algorithm for fault detection, diagnosis or recovery. This category of transitions can be exploded to provide a detailed representation of the operation it encapsulates.

A formal specification of the logical behavior of an agent's structure is accomplished using a top-down approach. This approach involves iteratively elaborating timed transitions to incorporate sufficient details of agent operations. Specifically, each timed transition is substituted with a more intricate and detailed sub-net, constructing an extensive PN, as depicted in Figure 36.

An essential factor in replacing a timed transition with a more complex sub-net is ensuring that the resultant PN preserves its properties of liveness, safeness, and boundedness, where liveness guarantees that the system does not reach a deadlock state, safeness ensures that no place contains more than one token at a time (preventing unintended overflows), and boundedness limits the maximum number of tokens in each place, ensuring controlled resource usage. Multiple theorems presented by Valette (1979) and later extended by Suzuki and Murata

Figure 36 – Example of transition explosion in the graphical Petri net formalism.



Source: (LEITÃO, 2004).

(1983) focus on the preservation of these characteristics within PN via stepwise refinement. Valette (1979) has shown that:

- Considering a transition t_a of a Petri net Z, a well-formed sub Petri net Z' and a Petri net Z'' as the result of the substitution of t_i by Z',
- The Petri net Z'' is: bounded if Z is bounded, safe if Z and the sub Petri net Z' are safe, and live if Z is live.

The theorem demonstrates that the method of transition refinement eliminates the need for analyzing the entire, extensive PN. Instead, all characteristics can be deduced from examining the initial PN and its associated sub-PNs. For comprehensive insights into PN theory and its mathematical foundations, see (MURATA, 1989; COLOMBO; CARELLI, 1997; VALETTE, 1979; CARDOSO; VALETTE, 1997).

4.2 AGENT DYNAMIC MODEL WITH CF* BEHAVIOR

The behavior modeling of each agent within the CPS, based on the CF* approach, is presented in this Section using the PN formalism. The proposed PN model captures the dynamic interactions between the physical and cyber components of the agent. This includes a loop for data processing for monitoring and controlling the physical component, parallel with the agent's

interaction, FT capabilities, and online agent models improvements. Furthermore, the model formalizes the collaborative interactions among agents required by the CF* approach, providing a structured and rigorous representation.

Initially, the graphical representation of the main PN is introduced, offering an intuitive depiction of the agent's behavioral progression. This model incorporates the essential structures necessary for implementing the CF*, including the control thread for the cyber-physical component, the interaction thread for communication with external agents, the FT thread addressing the detection, diagnosis, and recovery stages, and then the thread dedicated to improving FT models during the agent operation over time.

Finally, the mathematical model formalizing the proposed PN is described, encompassing the definitions of its places, transitions, input arcs, and output arcs, ensuring a robust and comprehensive representation of the agent behavior.

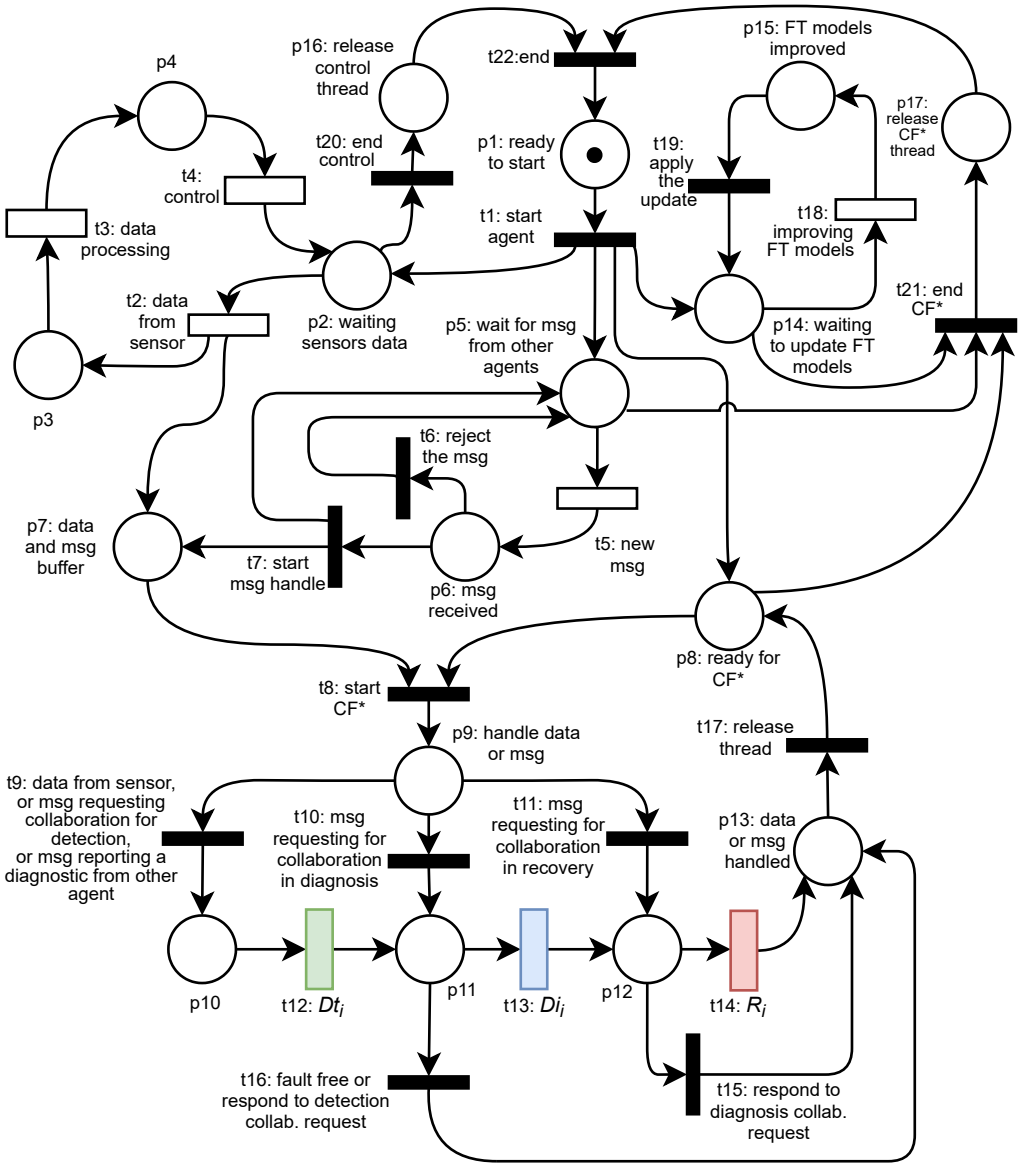
4.2.1 Fundamental Agent Behavior Modeled by Petri Net

Figure 37 presents the graphical representation of the proposed PN to model the behavior of an agent adopting the CF* approach. This PN (hereafter referred to as the fundamental PN or fundamental agent PN) captures the dynamic behavior of the agent, structured around the aforementioned four threads. These threads operate parallel to prevent mutual dependency or delays, ensuring efficient and uninterrupted agent performance.

The fundamental PN encapsulates the essential behaviors of the agent, managing the control loop layer to process sensor data for monitoring and controlling the physical component. Simultaneously, this data is routed through the FT stages, which can trigger collaborative requests in alignment with the CF* approach when necessary. Additionally, the model accounts for receiving and processing messages from other agents, enabling collaborative FT without compromising the agent's control processes. This parallel execution ensures that the collaboration mechanisms do not interfere with the core functionality of the agent.

As shown in Figure 37, the agent is ready to start its operation at $p1$, waiting for the start to trigger transition $t1$. After starting, the agent waits at $p2$ for the sensor data to control the physical component. In parallel, three tokens are created; the first is created in $p5$ and is responsible for receiving and processing messages from other agents regarding collaboration in fault detection, diagnosis, and recovery. The second token is created in $p8$ to manage the processing resources available to execute the CF*. To prevent loss of sensor data or collaborative

Figure 37 – Fundamental agent behavior modeled via PN integrating CF* approach.



messages caused by CF* processing, a buffer in $p7$ is used to store data and messages to be processed when $p8$ has an available token. The last token is generated at $p14$ dedicated to improving FT models during the agent operation over time. These tokens collectively represent the threads necessary for the agent's operation, created immediately after transition $t1$ is triggered.

The agent's dynamic behavior ends when transitions $t20$ and $t21$ are triggered, ending the control loop and CF* procedure, respectively; and then, transition $t22$ is triggered, ending the agent's system.

The timed transitions within the fundamental PN represent the more complex behaviors of the agent. These transitions can be exploded to provide a refinement, context-specific analysis. For example, transitions $t2$, $t3$, and $t4$ can be refined to detail the control loop behavior tailored to

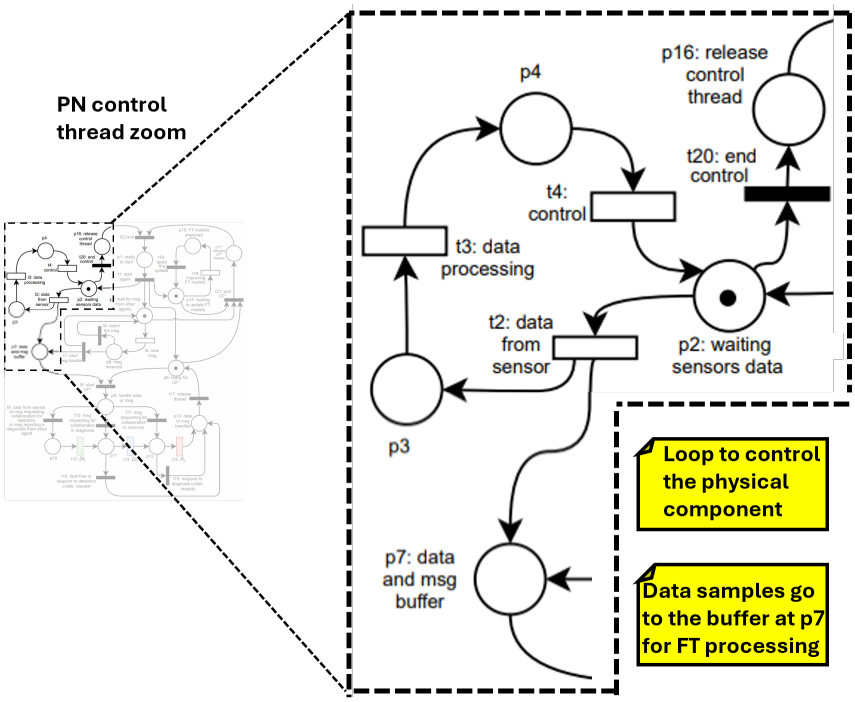
a specific CPS application. Similarly, transitions $t5$ and $t18$ can be expanded to describe, in more detail, the processes for receiving collaboration messages from external agents and for improving FT models, respectively. Further on in this Chapter, particularly in Section 4.3, transitions $t12$, $t13$, and $t14$ will be further expanded to present a detailed view of the collaborative behavior for each FT stage within the CF* approach.

Following, this Section will analyze the threads of the fundamental PN in more detail, focusing on control, interaction, fault tolerance stages, and model improvement.

4.2.2 Control Thread

Figure 38 presents an excerpt from the fundamental PN, focusing on the control thread. This thread begins at place $p2$, where it awaits the sampled data from the physical component, triggering transition $t2$ upon data availability. Transition $t3$ represents the pre-processing of this data, encompassing tasks, for example, data conditioning, monitoring, and visualization. Subsequently, transition $t4$ is responsible for controlling the physical component by calculating and transmitting the appropriate actions to component act and interact with the environment. This control loop corresponds to the control loop layer defined in the CF* meta-model (as illustrated in Figure 23).

Figure 38 – Excerpt of the fundamental PN referring to the control thread.



Whenever transition $t2$ is triggered, it produces tokens in two places. Firstly, place $p3$

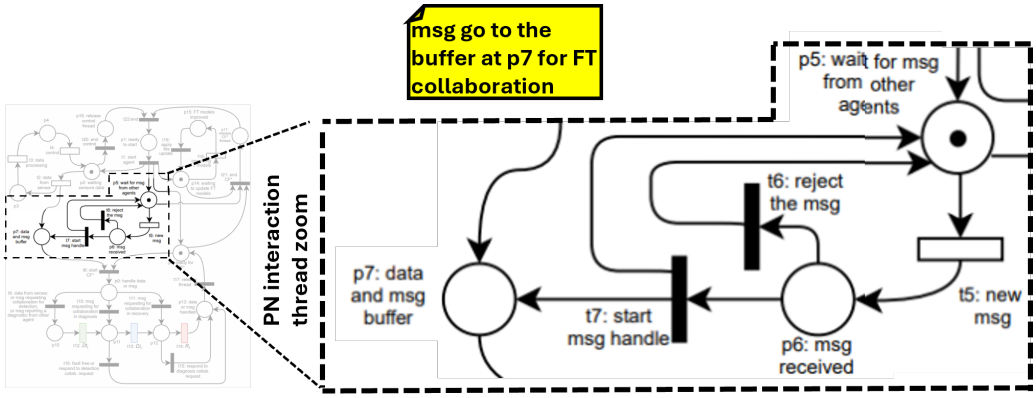
supports the continuation of the control loop by forwarding pre-processed data for further actions. Secondly, the place $p7$ acts as a temporary buffer for storing component sampled data. The token generated in $p7$ carries the context of the sampled data, ensuring that FT processing has information to operate effectively. This token is subsequently analyzed by the FT mechanisms, allowing the agent to execute detection, diagnostic, and recovery stages as needed.

The control thread concludes its operation when transition $t20$ is triggered, indicating the termination of the agent's control processes. This ensures the proper shutdown of the agent's operational cycle in a systematic and orderly manner.

4.2.3 Interaction Thread

The interaction thread, illustrated in Figure 39, plays a pivotal role in the CF* approach. This thread begins at place $p5$, where it waits to receive messages from external agents. These messages typically involve requests for collaboration in one of the FT stages, such as detection, diagnosis, or recovery. When the transition $t5$ is triggered, the agent receives a new message. Based on the agent's availability or compatibility with the external agent, the message can either be accepted or rejected, triggering transitions $t7$ or $t6$, respectively.

Figure 39 – Excerpt of the fundamental PN referring to the interaction thread.



A token is added to place $p7$ when a collaboration request is accepted. This token encapsulates the context of the received message, including the type of collaboration requested. For instance, if a detection request is accepted, the token will carry the necessary information and associated data to be analyzed by the agent. Once this token has been processed by one of the FT stages, the local agent responds to the external agent.

This interaction thread aligns with the interactions and communication mechanisms defined in the CF* meta-model (as depicted in Figure 23), emphasizing the collaborative nature

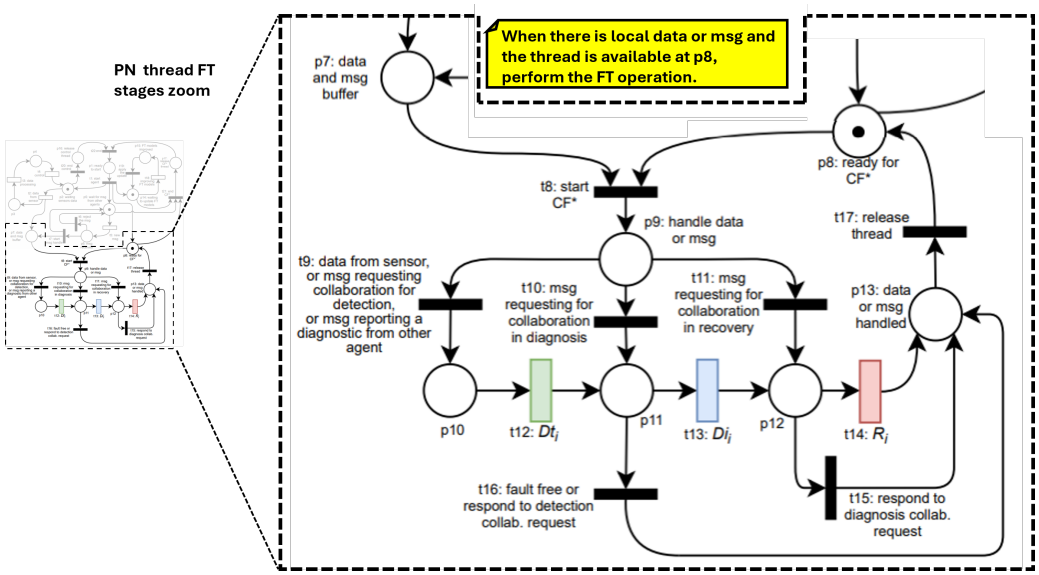
of the CF* approach.

4.2.4 Fault Tolerance Thread

The FT thread, depicted in Figure 40, is responsible for analyzing data from both the control loop and collaborations with other agents. This thread begins at place $p8$ and operates based on the availability of tokens in the data and message buffer, $p7$. The processing sequence can follow, for example, a First in first out (FIFO) logic or prioritize tokens based on specific criteria, such as urgency or relevance, depending on the CPS application. Although more sophisticated methods incorporating prioritization can be adopted to manage the buffer, to facilitate the explanation and clarity, the FIFO logic will be used to explain the operation.

This thread aligns with the fault tolerance loop layer of the CF* meta-model (refer to Figure 23), emphasizing its central role in ensuring robust and collaborative FT.

Figure 40 – Excerpt of the fundamental PN referring to the fault tolerance thread.



When tokens are present in $p7$ and $p8$, the $t8$ transition is triggered. The place $p9$ is specific for handling the transitions of the different CF* stages, indicating which stage the agent will execute. Three transition possibilities exist in this step, and the decision is made based on the token context. For example, suppose that the token at $p7$ originates from the data received from the sensor of the physical component. In this case, performing an inference in the fault detection stage is necessary; therefore, transition $t9$ is triggered. After $p10$, transition $t12$ is triggered to detect the fault. If a fault is detected at $t12$, a diagnosis is made with transition $t13$ and recovery at $t14$. However, if $t12$ does not detect a fault, there is no need for diagnosis and

recovery; therefore, the token in $p11$ will perform the sequence $[t16,p13,t17,p8]$, where the agent will be able to process another token that is in $p7$. The same situation occurs when another agent reports a diagnosis (Di_j^*), and the local agent must check whether this diagnosed fault affects it.

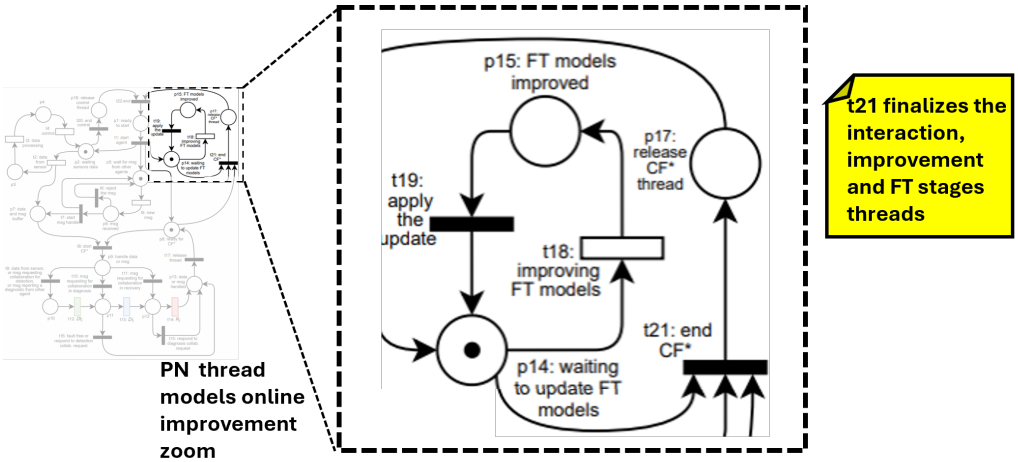
However, if the token context in $p7$ originates in a message from another agent requiring collaboration for any of the CF* stages, an appropriate transition is triggered when the token is in $p9$. For example, if the agent receives a message requesting collaboration for the diagnosis, on this occasion, transition $t10$ is triggered, moving the token to $p11$, then executes transition action $t13$ responding to the collaboration request through the sequence $[t10,p11,p12,t15,p13,t17,p8]$. If, for example, the context of the token in $p9$ is a request for collaboration in detection, the sequence will be $[t9,p10,t12,p11,t16,p13,t17,p8]$; in case of a collaboration request for recovery, the sequence will be $[t11,p12,t14,p13,t17,p8]$.

The timed transitions $t12$, $t13$, and $t14$ represent the stages of fault detection, diagnosis, and recovery, respectively. If the token originates from a collaboration request, these transitions also facilitate a response to the initiating agent, thereby closing the loop of interaction and analysis. These transitions will be expanded further in subsequent Sections of this Chapter to provide a more detailed view of their operation and refinement.

4.2.5 Models Improvement Thread

The model improvement thread, illustrated in Figure 41, is responsible for the online enhancement of fault detection, diagnosis, and recovery models of the CF*. This thread aligns with the system model layer of the CF* meta-model (refer to Figure 23), ensuring continuous adaptation and refinement of the FT mechanisms.

Figure 41 – Excerpt of the fundamental PN referring to the models improvement thread.



The thread begins at place $p14$ and is triggered whenever a model update is required. This necessity may arise, for instance, due to a timeout event or the identification of deficiencies in the model by metrics derived from the 7CM. In such cases, the timed transition $t18$ is activated. Although $t18$ encapsulates complex behaviors that could be expanded to define specific update procedures, these details fall outside the scope of this thesis.

Transition $t19$ ensures updated FT models are integrated into the respective FT mechanisms, enabling the system to operate with improved accuracy and efficiency.

Finally, transition $t21$ concludes the operation of the CF* approach when the interaction, fault tolerance, and model improvement threads are ready to finalize. Notably, the four threads discussed operate parallel, ensuring no operation is concurrent or delayed and resource contention is effectively avoided.

4.2.6 Mathematical Modeling of the Fundamental Agent Behavior PN

The PN used to model the general behavior of the agent in the CPS with the CF* approach is formally defined as a tuple $PN = \{P,T,I,O\}$. This formalization guarantees an unequivocal and precise depiction of the agent's components and their connections, conforming to the mathematical rigor requisite for the analysis and validation of the proposed model.

The P represents the set of places, serving as containers for tokens, which indicate that a task is ready for execution, a resource is available, or a specific state of the agent has been reached. The places P for the PN of agent with CF* is represented in the Equation (11):

$$P = \{p1,p2,p3,p4,p5,p6,p7,p8,p9,p10,p11,p12,p13,p14,p15,p16,p17\} \tag{11}$$

The T denotes the set of transitions, serving as the bridge between preconditions (represented by input places) and post-conditions (represented by output places). The T model the activities that trigger changes between places in the PN, represented in Equation (12):

$$T = \{t1,t2,t3,t4,t5,t6,t7,t8,t9,t10,t11,t12,t13,t14,t15,t16,t17,t18,t19,t20,t21,t22\} \tag{12}$$

The I is the set of input arcs in PN transitions, where $I = (P \times T)$, mapping the arcs from places (rows) to transitions (columns), thereby defining the preconditions for each transition to occur. As established by the Equation (13), for the developed PN, each transition consumes precisely one token from each preceding place. For example, the transition $t1$ consumes one token from place $t1$, or the transition $t8$ consumes one token in $p7$ and one token in $p8$.

$$
I = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix} \tag{13}
$$

The O is the set of output arcs in PN transitions, where $O = (P \times T)$, mapping the arcs from transitions (columns) to places (rows), thereby defining the pos-conditions after each transition occurrence. As established by the Equation (14), for the developed PN, each transition provides precisely one token from each following place. For example, the transition $t1$ provide one token to the places $p2$, $p5$, $p8$ and $p14$.

$$
O = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{bmatrix} \tag{14}
$$

To define the PN initialization, the m_0 represented in Equation (15) specifies the distribution of tokens across the places P at the beginning of the agent's operation, determining which transitions are enabled at the start.

$$m_0 = \{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0\} \tag{15}$$

4.3 PETRI NETS REFINEMENT OF THE FAULT TOLERANCE BEHAVIOR

The fundamental PN, shown in Figure 37, models the high-level behavior of an agent utilizing the CF* approach. It includes timed transitions within various threads that encapsulate complex operations. These transitions can be refined to explore specific aspects of the agent's behavior in enhanced detail.

While all expandable transitions significantly impact the agent's operation, this thesis

focuses on refining those associated with the FT stages. Specifically, the transitions $t12$ (collaborative fault detection), $t13$ (collaborative fault diagnosis), and $t14$ (collaborative fault recovery) will be expanded. These refinements aim to provide a detailed representation of how FT is achieved both locally and collaboratively, adhering to the principles of the CF* approach and the interaction protocols described in Section 3.5.

As an alternative to the dynamic modeling with PN for the FT stages in the CF* approach, Appendix C presents flowcharts that outline the behavior of these stages. These flowcharts serve as supplementary material to aid in understanding the PN models introduced in this Section in case more elucidation is necessary.

4.3.1 Refinement of Collaborative Fault Detection Transition

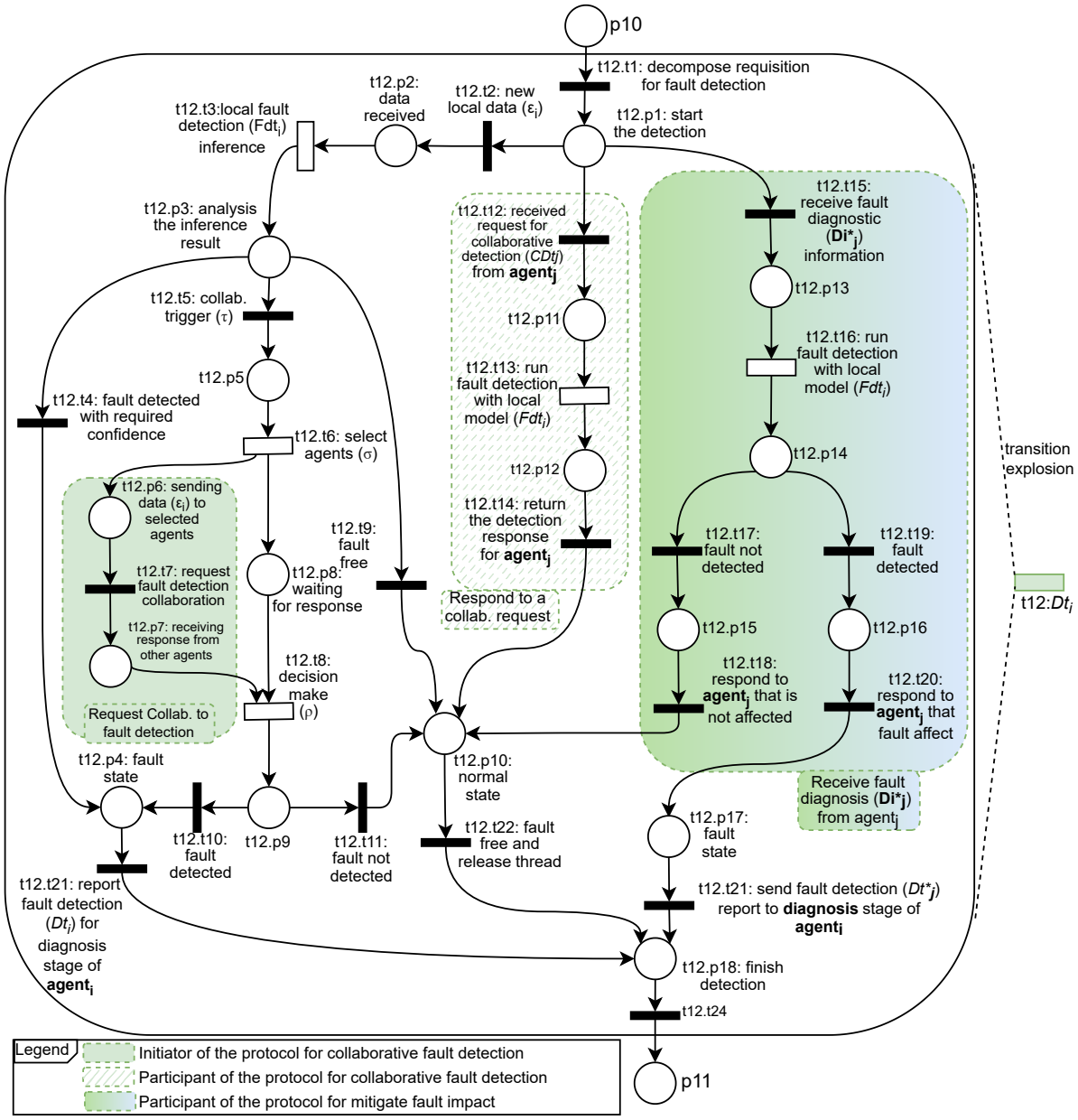
The initial stage of the FT process, i.e., the fault detection stage, was previously specified and discussed in Section 3.4.2. This stage is modeled by a PN that refines transition $t12$ of the fundamental PN.

Figure 42 shows the PN refinement resulting from the explosion of transition $t12$, referred to as the collaborative fault detection PN. This PN represents the dynamic behavior of the agent during the fault detection stage in the CF* approach. Four main capabilities characterize the dynamic fault detection behavior:

- The local ability to detect faults when possible;
- The ability to request collaboration when fault detection uncertainty occurs (transitions and places highlighted in solid green);
- The ability to respond to a collaboration request for fault detection (represented by transitions and places highlighted with dashed green) by analyzing the provided data with the local fault detection model;
- The ability to assess whether a fault reported by another agent may affect the local agent (represented by transitions and places highlighted in gradient green-blue).

Initially, in transition $t12.t1$, the request for fault detection is managed according to the token context when available in place $p10$. Suppose the context refers to new sensor data, the $t12.t2$ transition prepares the data; then, transition $t12.t3$ performs the inference according to

Figure 42 – PN refinement modeling the dynamic of collaborative fault detection.



the established Fdt_i fault detection model. In place $t12.p3$, there are three different sequences of fault detection possibilities:

- *Fault Detected Locally*: The first sequence occurs when a fault is detected without activating the τ collaboration criteria. In this situation, transition $t12.t4$ is triggered, resulting in a fault state in place $t12.p4$, to later report the detected fault to the diagnosis stage according to the sequence $[t12.t21, t12.p18, t12.t24, p11]$;

- *No Fault Detected*: The second possibility occurs when the analyzed data is unrelated to a fault, i.e., fault-free. Transition $t12.t9$ is triggered, resulting in a normal operating state

($t12.p10$), followed by the sequence [$t12.t22$,$t12.p18$,$t12.t24$,$p11$];

- *Collaboration Triggered*: The third possibility begins if the collaboration trigger τ is activated (e.g., due to uncertainty or other criteria discussed in Section 3.2.2), and transition $t12.t5$ is fired. Transition $t12.t6$ then selects agents for collaboration based on parameter σ. After exchanging data per the fault detection collaboration protocol[1], the local agent evaluates the collaboration results in transition $t12.t8$ using parameter ρ. If a fault is detected, the sequence [$t12.p9, t12.t10, t12.p4, t12.t21, t12.p18, t12.t24$,$p11$] updates the agent's state to fault and reports it to the diagnosis stage.

However, the trigger on $t12.t1$ can be performed by a token with a context requiring collaboration. In this case, after triggering $t12.t1$, the token is placed at $t12.p1$ triggers transition $t12.t12$, starting the sequence to respond to a collaboration request. Transition $t12.t13$ executes the fault detection model of local agent Fdt_i using the data from the agent that initiated the collaboration. In the transition $t12.t14$, the inference result is returned along with the model accuracy and precision, ending with the sequence [$t12.p10, t12.t22, t12.p18, t12.t24, p11$].

Finally, the token that triggers on $t12.t1$ may have a fault report context generated by the diagnosis stage of another agent. In this case, after triggering $t12.t1$, token in place $t12.p1$ triggers transition $t12.15$, which receives a diagnosis from another agent. If no fault is detected, the agent's operating state remains normal. Otherwise, a fault state is assigned before proceeding to the diagnosis stage.

4.3.2 Refinement of Collaborative Fault Diagnosis Transition

After a fault is detected, it is necessary to determine its cause and impact on the FT process. The fault diagnosis stage was previously specified and discussed in Section 3.4.3. This stage is modeled by a PN that refines the transition $t13$ of the fundamental PN.

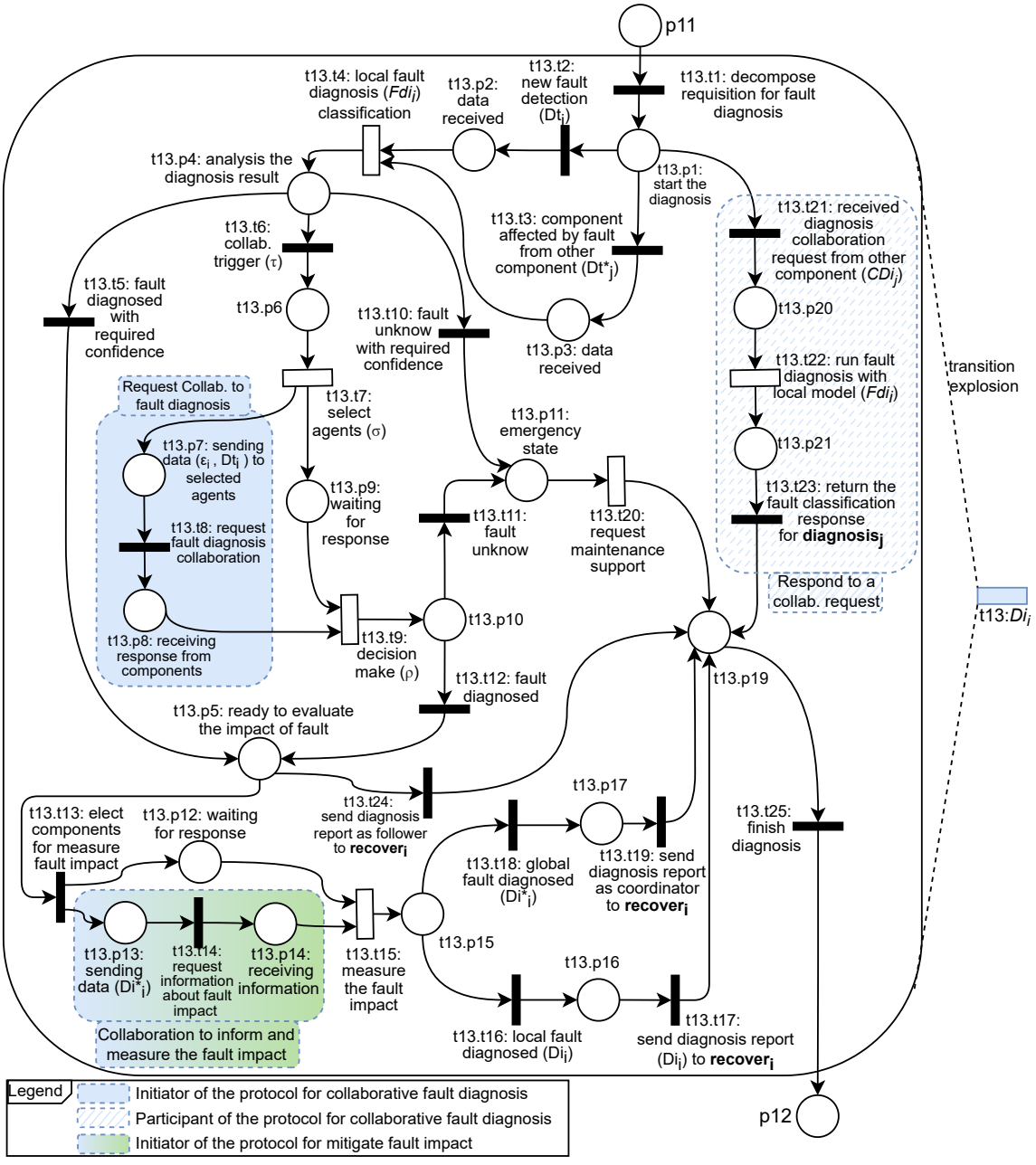
Figure 43 shows the PN refinement resulting from the explosion of transition $t13$, named the collaborative fault diagnosis PN. This PN represents the dynamic behavior of the agent during the diagnosis stage of the CF*. The diagnosis dynamic behavior model encompasses five main capabilities:

- The local ability to diagnose faults when possible;

[1] The collaborative fault detection protocol is presented earlier in Figure 30.

- The ability to request collaboration when fault diagnosis uncertainty occurs (transitions and places highlighted in solid blue);
- The ability to respond to a diagnosis collaboration request (transitions and places highlighted with dashed blue) by analyzing the provided data with its local model;
- The ability to check and mitigate the fault impact, i.e., to verify whether the diagnosed fault has a local or global impact (transitions and places highlighted in gradient blue-green);
- The ability to request maintenance support when the fault cannot be diagnosed.

Figure 43 – PN refinement modeling the dynamic of collaborative fault diagnosis.



Initially, in transition $t13.t1$, the request for fault diagnosis is managed according to the token context when available in place $p11$. Suppose the context refers to a new fault detection episode (provided of previews transition $t12$ of PN fundamental), the $t13.t2$ transition prepares the data; then, transition $t13.t4$ performs the inference according to the established Fdi_i fault diagnosis model. In place $t13.p4$, there are three different sequences of fault diagnosis possibilities derived from the analysis of the diagnosis result:

- *Local Diagnosis*: The first possibility occurs when a fault is diagnosed outside the τ collaboration criteria, i.e., the agent can diagnose the fault locally. In this situation, transition $t13.t5$ is triggered, and therefore, the agent is ready to start mitigating the impact of the fault diagnosed at place $t13.p5$;

- *Collaborative Diagnosis*: The second possibility at the place $t13.p4$ derived from the analysis of the diagnosis result, starting when the collaboration trigger (τ) is fired on transition $t13.t6$, i.e., the agent needs collaboration to diagnose the fault. Then, in transition $t13.t7$, the agents for collaboration are selected according to parameter σ. After exchanging data for the collaboration according to the fault diagnosis protocol[2], $[t13.p7, t13.t8, t13.p8]$ the local agent, in transition $t13.t9$ evaluates the collaborations and decides according to parameter ρ. If the decision results in an appropriate fault diagnosis, the sequence $[t13.p10, t13.t12, t13.p5]$ follows, and the agent is ready to start mitigating the impact of the fault at place $t13.p5$. However, maintenance support will be requested to diagnose the fault if the fault remains undiagnosable, according to the flow $[t13.t11, t13.p11, t13.t20, t13.p19, t13.t25, p12]$;

- *Unknown Fault*: The third possibility for diagnosis at the place $t13.p4$ occurs when the fault is unknown locally with the required confidentiality, represented by the transition $t13.t10$, and therefore, initiates the procedure to request maintenance support through the flow $[t13.t10, t13.p11, t13.t20, t13.p19, t13.t25, p12]$;

After a fault has been properly diagnosed, the agent must mitigate its impact in the context of CF*. This behavior starts at place $t13.p5$, if the fault was diagnosed by an external agent (according to the initial transition at $t13.t3$), the agent will assume the follower role, receiving orders from the external agent that diagnosed the fault according to the sequence $[t13.p5, t13.t24, t13.p19, t13.t25, p12]$.

[2] Protocol for collaborative fault diagnosis described previously in Figure 31

Conversely, if the local agent was the first to diagnose the fault, the mitigation process is carried out by interacting with the detection stage of the external agents, according to flow $[t13.t13, t13.p13, t13.t14, t13.p14, t13.t15]$. This information is transmitted on to the recovery stage if the fault is local, according to flow $[t13.p15, t13.t16, t13.p16, t13.t17, t13.p19, t13.t25, p12]$. However, if the fault is global, i.e., affects more agents in the system, the local agent must assume the coordinator role in the recovery stage, according to the flow $[t13.p15, t13.t18, t13.p17, t13.t19, t13.p19, t13.t25, p12]$.

Finally, another agent behavior occurs when it receives a request to collaborate, i.e., the trigger on $t13.t1$ can be performed by a token with a context requiring collaboration, in which case, after triggering $t13.t1$, the token in place $t13.p1$ triggers transition $t13.t21$, starting the sequence to respond to a collaboration request. Transition $t13.t22$ executes the fault diagnosis model of local agent Fdi_i using the data from the agent that initiated the collaboration. The diagnosis result is returned in the transition $t13.t23$, ending according the sequence $[t13.p19, t13.t25, p12]$.

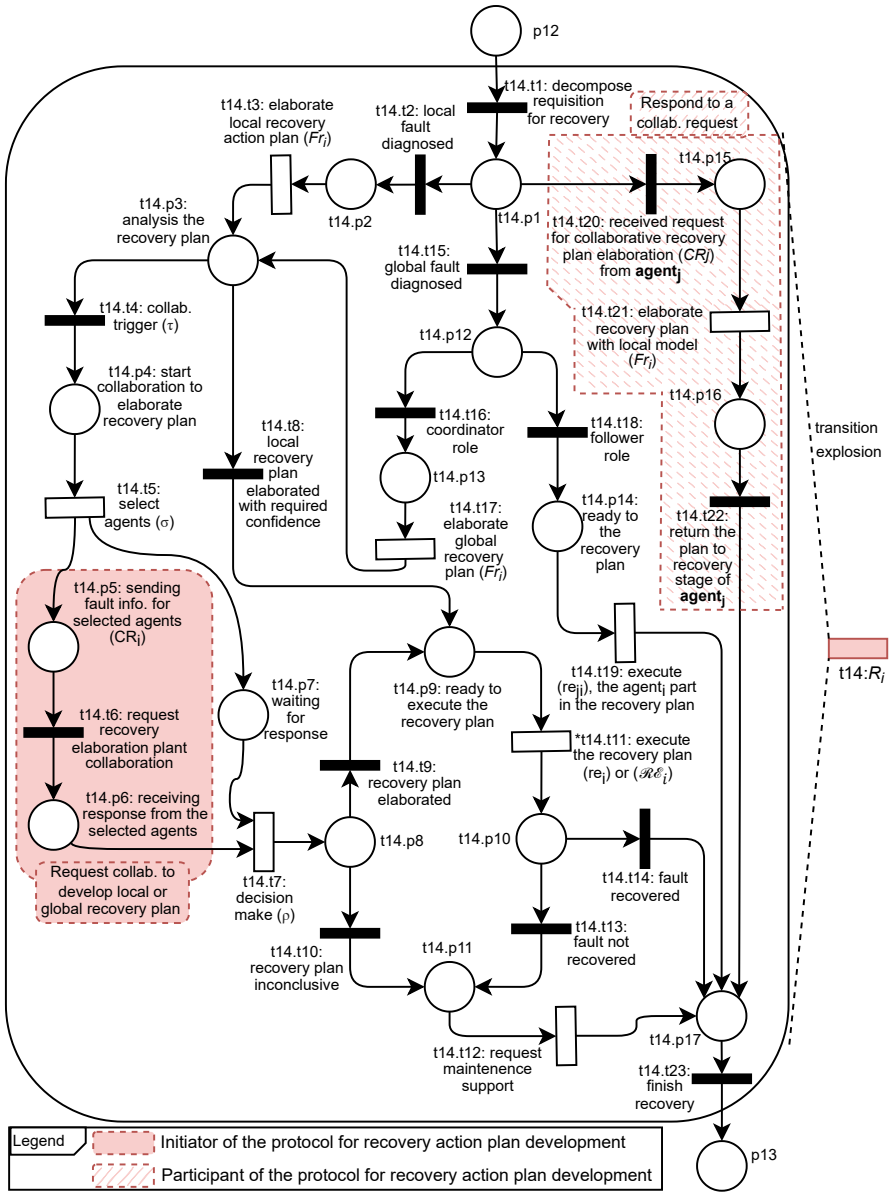
4.3.3 Refinement of Collaborative Fault Recovery Transition

The transition $t14$ in the fundamental PN encapsulates the final stage in the FT process, precisely the fault recovery stage. This stage, previously specified and discussed in Section 3.4.4, has its dynamic behavior modeled by a PN resulting from the refinement of $t14$.

Figure 44 illustrates the PN refinement resulting from the expansion of transition $t14$, termed as the collaborative fault recovery PN. The recovery dynamic behavior model encompasses five main capabilities:

- The local ability to elaborate the recovery action plan when possible;
- The ability to request collaboration when unable to develop an adequate recovery action plan (transitions and places highlighted in solid red);
- The ability to respond to a recovery collaboration request (transitions and places highlighted with dashed red);
- Coordinate a fault recovery with global impact, i.e., involving more than one agent;
- Follow an external coordinator's recommendations to recover from a fault with a global impact;

Figure 44 – PN refinement modeling the dynamic of collaborative fault recovery.



Initially, in transition $t14.t1$, the request for fault recovery is managed according to the token context available in place $p12$. Three different contexts are modeled: fault locally diagnosed, fault globally diagnosed, and responding to a collaboration request to develop the recovery action plan.

In the first context, when a local fault is diagnosed, transition $t14.t2$ is initially triggered; then, transition $t14.t3$ elaborates the local recovery action plan according to the established Fr_i fault recovery model. In place $t14.p3$, two different sequences of fault recovery possibilities are derived from the analysis of the recovery action plan result. The first occurs when the action recovery plan is outside the τ collaboration criteria, i.e., the agent can elaborate the recovery plan locally. In this scenario, transition $t14.t8$ is triggered, and the agent is ready to

execute the recovery plan at place $t14.p9$. The second possibility at place $t14.p3$ arises when the collaboration trigger (τ) is fired on transition $t14.t4$, i.e., the agent needs collaboration to elaborate the recovery plan. Then, in transition $t14.t5$, the agents for collaboration are selected according to parameter σ. After exchanging data for the collaboration according to the protocol[3] for collaborative recovery action plan elaboration, [$t14.p5$,$t14.t6$,$t14.p6$], the local agent, in transition $t14.t7$ evaluates the collaborations and defines a plan according to parameter ρ. If the decision results in an appropriate recovery plan, the sequence $[t14.p8, t14.t9, t14.p9]$ follows, and the agent is ready to execute the recovery plan at place $t14.p9$. The execution of the local recovery action plan occurs in transition $t14.t11$, executing the local recovery action re_i. Then, in place $t14.p10$, it is verified whether the recovery was successful, which will follow the sequence $[t14.p10, t14.t14, t14.p17, t14.t23, p13]$ or unsuccessful, requiring external support $[t14.p10, t14.t13, t14.p11, t14.t12, t14.p17, t14.t23, p13]$. However, if it is not possible to formulate a recovery plan, maintenance support will be requested to recover from the fault, according to the flow $[t14.t10, t14.p11, t14.t12, t14.p17, t14.t23, p13]$.

In the second context related to the initial transition $t14.t1$, when a global fault is diagnosed, transition $t14.t5$ is triggered. In this circumstance, the agent can assume two roles: coordinator or follower. If the agent is the coordinator of the fault recovery, the sequence $[t14.t15, t14.p12, t14.t16, t14.p13, t14.t17, t14.p3]$ will be triggered, where the global recovery plan $\mathcal{RE}i$ will be developed, and the same behavior observed in the first context (when the fault is local) is performed. The differentiation occurs in transition $t14.t11$, where the agent will coordinate the recovery plan. However, if its role is that of a follower, the sequence $[t14.p12, t14.t18, t14.p14, t14.t19, t14.p17, t14.t23, t14.t23, p13]$ will be executed, where, in transition $t14.t19$, the agent executes the recovery action re_{ji} coordinated by the external $agent_j$. For example, consider two autonomous robots, A and B, operating in a warehouse. When a global path fault occurs, robot A assumes the coordinator role, analyzes the fault, and devises a recovery plan, instructing robot B to update and adapt its route. As the follower, Robot B executes these instructions to resume its operations.

Finally, another agent behavior occurs when it receives a request to collaborate in elaborating the recovery action plan, i.e., the trigger on $t14.t1$ can be performed by a token with a context requiring collaboration, in which case, after triggering $t14.t1$, the token in place $t14.p1$ triggers transition $t14.t20$, starting the sequence to respond to a collaboration request.

[3] Protocol for collaborative recovery action plan elaboration described previously in Figure 33

Transition $t14.t21$ executes the local agent's Fr_i model using the data from the agent that initiated the collaboration. In transition $t14.t22$, the plan is returned, ending with the sequence $[t14.p17, t14.t23, p13]$.

4.4 FORMAL ANALYSIS OF FUNDAMENTAL PETRI NET OF CF*

The employment of PN models, firmly rooted in mathematical principles, such as functional analysis and linear algebra, enables a rigorous evaluation of key PN properties. These properties, essential for the structural and behavioral properties of the PN, often referred to as qualitative analysis, include resource boundedness, reversibility, deadlock-freeness, resource conservativeness, and safety. Additionally, the performance analysis of the PN, often referred to as quantitative properties, is evaluated using discrete-event simulation techniques. This dual analysis approach provides a comprehensive understanding of both the qualitative and quantitative characteristics of the fundamental PN model and provides insights into the robustness and adaptability of the CF* model for CPS.

For simulation and validation, the Petri nets development toolKit (PndK) software (LEITÃO *et al.*, 2009) will be employed. This software facilitates reconfigurable and modular systems' specification, design, analysis, validation, and control through PN. Developed in the C++ programming language, PndK supports key functionalities such as editing, structural and behavioral analysis, and performance simulation of the PN models (LEITÃO *et al.*, 2009).

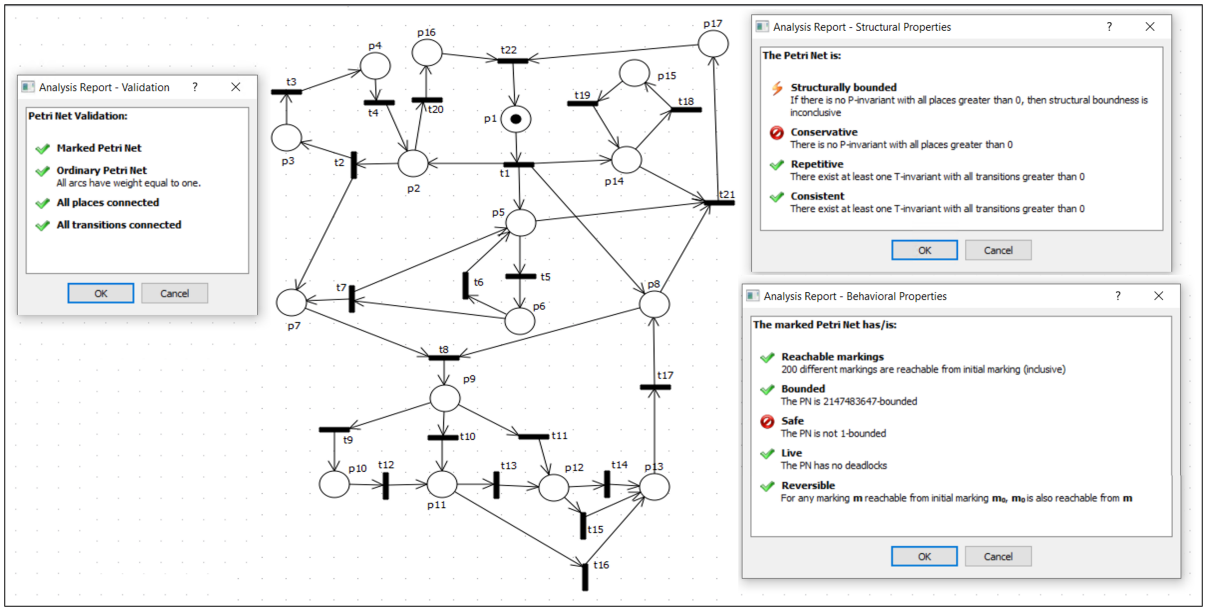
In this Section, the results of qualitative and performance analyses will be demonstrated using the fundamental PN, which models the general behavior of the agent with CF*.

Figure 45 showcases the configuration of the fundamental PN within the PndK software, highlighting its role in the structural and behavioral properties analysis for the validation of CF*.

4.4.1 Structural Analysis

The analysis of structural properties, illustrated in Figure 45, confirms that the PN model is consistent and repetitive, representing a cyclical sequence of transition triggers that return the system to its initial marking, m_0, while maintaining the overall conservation of the system. This analysis implies that the agent can sustain continuous operation without encountering deadlocks or unreachable places, ensuring that all structural elements of the PN actively participate in the behavior of the agent. The consistency property guarantees that every transition contributes to

Figure 45 – Structural and behavioral analysis of the PN of the agent with CF* approach.



at least one cycle of activity, reflecting the operational completeness of the model. Similarly, the repetitiveness property ensures that the PN can repeatedly execute its modeled behaviors, a critical requirement for the agent's ongoing processes such as control, monitoring of cyber-physical components, and FT collaboration within the CF* approach.

However, the structural properties analysis reveals that the PN is not conservative, indicating that tokens are created and destroyed throughout the operational cycles. This analysis indicates that the fundamental PN does not conserve tokens across all places. For the agent behavior, this reflects the agent's ability to generate new tokens as new data from components are digitized for control or new collaboration messages become available for processing. These tokens are generated in place $p7$, representing the data and message buffer of the fundamental PN (discussed previously in Figure 37). They are consumed by transition $t8$, which activates when the FT processing thread (represented by place $p8$) becomes available.

As a result of this desirable token creation and consumption characteristic at the buffer, the property of structural boundedness is inconclusive. While the PN may exhibit bounded behavior under specific initial markings, its structure alone does not guarantee boundedness in scenarios where place $p7$ could store an unbounded number of tokens due to an influx of messages and data. This inconclusiveness does not imply that the agent is inherently uncontrolled. Instead, it highlights the need for careful synchronization between the control and fault tolerance threads. Specifically, the control thread should not execute significantly faster than the fault tolerance thread, and the acceptance of collaboration requests must be intelligently considered to

ensure this synchronization.

In summary, to prevent unbounded token accumulation in place $p7$, two key considerations should be addressed:

- *Synchronization between control and fault tolerance cycles*: The execution cycle of the control thread should align with the fault tolerance thread's processing capacity. If the control cycle is significantly faster, the arc connecting transition $t1$ to place $p8$ could have a higher weight, ensuring that more tokens are processed in $p8$ to match the rate of data generation.

- *Buffer management*: If the buffer (place $p8$) accumulates too many tokens, the agent should reject additional collaboration requests to avoid overloading or handling outdated in the fault tolerance thread.

For example, consider an AMR equipped with LiDAR sensors to map and navigate an environment while avoiding obstacle collisions. This sensor continuously scans data to control the robot and simultaneously stores it temporarily in a buffer (analogous to place $p7$). If the robot's FT system processes this data more slowly than the LiDAR generates it, the buffer may grow indefinitely, leading to delays in fault detection and diagnosis, ultimately compromising the AMR's safe and free collision navigation. To mitigate this, the agent responsible for the AMR must synchronize the control cycle with the FT cycle. The robot can discard older data or reject new collaboration requests, adopting policies that refuse lower-priority requests when the buffer reaches a critical limit. This situation directly illustrates the impact of an inconclusive boundedness analysis: while the PN structure does not guarantee explicit limits, proper control can prevent practical issues.

Table 10 summarizes the structural properties analysis of the PN that models the fundamental PN of the agent with CF* approach.

Table 10 – Summary of the structural properties of the fundamental PN for an agent with CF* approach.

Structural Property	**Result**	**Meaning for the agent represented by the PN**
Repetitive	Yes	The agent can execute repeatedly its model without deadlocks
Consistence	Yes	The agent model has no inactive transitions and can return to initial states
Conservative	No	Tokens are created and destroyed specifically in $p7$, indicating the existence of local data or messages from other agents to be processed for FT.
Structurally Bounded	Undetermined	The amount of tokens in position $p7$ depends on the amount of local data or messages to be processed.

4.4.2 Behavioral Analysis

The analysis of the behavioral properties use linear algebra methods based on the equations presented in Section 4.2.6 and is supported by the PndK software tool. These properties assess the dynamics of token movement through the places and transitions of the fundamental PN, providing essential insights into the functionality and robustness of the model.

As obtained using the software (illustrated in Figure 45), the developed fundamental PN is characterized by reachability, with 200 distinct markings accessible from the initial marking m_0 (inclusive). This result is valuable for evaluating the system's behavior in terms of all its potential states.

Additionally, the analyzed fundamental PN exhibits the property of liveness, meaning it is a live PN and free of deadlocks. Practically, liveness ensures that the agent executes all its threads, namely, those related to component control, collaboration, model improvement, and data analysis for FT, without encountering situations where execution stops due to unresolved dependencies.

Reversibility is another critical property observed in the fundamental PN. From any reachable marking, it is possible to return to the initial marking m_0 through a sequence of transitions. This characteristic guarantees that the modeled agent can correctly initialize and terminate within the CPS.

Finally, the remaining properties, safety, and boundedness, complement the analysis. The software results indicate that the PN does not exhibit the safety property. This property implies that during execution, a specific place, $p7$, can hold more than one token simultaneously. This behavior is intentional, as $p7$ is designed to act as a buffer for local data and external agents' messages. This feature allows the PN to temporarily store information in $p7$ without losing data or disrupting synchronization locally and in collaborative operations.

Table 11 summarizes the behavioral analysis of the PN that models the fundamental PN of the agent with CF* approach.

Table 11 – Summary of the behavioral properties of the fundamental PN for an agent with CF* approach.

Behavioral Property	Result	Meaning for the agent represented by the PN
Reachable	Yes	The PN presents a moderate diversity of accessible states
Live	Yes	The PN do not present deadlocks to execute the control loop and FT concurrently
Reversible	Yes	PN can be started and ended correctly
Bounded and Safe	No	The buffer at position $p7$ can store data and information to be processed when there is a thread available.

4.4.3 Performance Analysis

As with the structural and behavioral analysis, the performance analysis is focused solely on the fundamental PN. This analysis introduces the concept of time, which is associated with each transition of the PN discussed previously. To facilitate this, a deterministic time distribution is adopted to represent the evolution of the PN and evaluate its performance.

The performance analysis of the fundamental PN utilizes abstract time units, referred to as "ticks" to represent the relative duration of the PN transitions. These units are intentionally detached from temporal measurements (e.g., milliseconds or seconds) to represent the relative time consumed and provide a platform-agnostic and implementation-independent approach. This enables the performance evaluation to be broadly applicable across various CPS components and scenarios, independent of hardware or software constraints.

Three distinct scenarios are presented to explore the dynamic behavior of the CF* under different operational conditions. These scenarios illustrate key behavioral properties, such as synchronization, safety, and boundedness, under varying agent operations configurations. The scenarios are defined as follows:

- *(Scenario #1) Synchronized Local Control and Local FT Thread*: This scenario represents the simplest operational mode, where the control thread is synchronized with the FT thread, considering just the local model of the agent with CF* approach, without collaboration and FT models improvement.

- *(Scenario #2) Synchronized Control, FT, Collaboration and Model Improvement Thread*: This scenario introduces additional complexity by incorporating threads for collaboration requests and model improvement. These threads operate alongside the control and FT threads while maintaining synchronization.

- *(Scenario #3) De-synchronized operation*: This scenario examines the impact of de-synchronization between the control and FT threads, simulating a configuration where the control thread operates at a fast pace than the FT thread.

Table 12 summarizes the tick values assigned to each transition in the fundamental PN for the three scenarios. These values provide the foundation for simulating the agent's performance and analyzing the impact of ticks on its behavior. A deterministic approach is

employed for the tick assignments due to limitations of the PndK software tool, which does not support stochastic time or time variability driven by functions or events.

Table 12 – Ticks associated with the transitions for the performance analysis.

Transitions	Associated Thread	Ticks for Scenario #1	Ticks for Scenario #2	Ticks for Scenario #3
t2, t3, t4	Control	5	5	1
t5	Collaboration	Not triggered	12	Not triggered
t12, t13, t14	FT	2	2	2
t18	Model improvement	Not triggered	48	Not triggered
Remaining transitions	All threads	1	1	1

For temporized transitions, namely transitions $t2, t3, t4, t5, t12, t13, t14,$ and $t18$, which represent more computationally intensive or multi-step processes, an increased allocation of ticks is assigned. For the remaining transitions, which represent basic operations, the assigned time is set to 1 tick. These transitions involve minimal processing or state changes and, therefore, have the shortest execution time.

Scenario #1 focuses on the local control and FT operation of an agent modeled with the fundamental PN. To simplify the analysis and aid comprehension, collaboration and model improvement behaviors are disabled in this scenario, meaning transitions $t5$ and $t18$ are not fired. The scenario includes two initial fault detections, indicating that the system is fault-free. Subsequently, a fault is introduced, triggering fault detection, diagnosis, and recovery. Finally, another fault detection indicates that the system has returned to a fault-free state. Then, the agent operation based on the PN process is concluded, returning the agent to its initial state.

Figure 46 illustrates the evolution of the fundamental PN for scenario #1[4], along with the consumption of the ticks for each transition. Based on the assigned timings for each transition, the control thread completes its cycle at every 15 ticks. After completing transition $t2$, which digitizes the data, the transition $t8$ is fired to initiate the FT thread. The fault detection is performed between ticks 6 and 10, during which transition $t12$ is active. Since no fault is detected in this cycle, the transitions $t13$ and $t14$ are not fired; instead, transition $t16$ is triggered, indicating that the system is fault-free. This process is repeated between the ticks 21 and 26, as well between the ticks 51 and 56.

However, when a fault is detected, the agent proceeds through fault detection, diagnosis, and recovery, as observed between ticks 36 and 45. During this period, transitions $t12$, $t13$, and $t14$ are fired sequentially. The threads conclude their processes when transitions $t20$ and $t21$ are fired, followed by transition $t22$, which resets the agent to its initial state.

[4] Check the video of PN for scenario #1 on https://youtu.be/izDYfZ0zojU

Figure 46 – Gantt diagram for performance analysis of the fundamental PN for scenario #1.

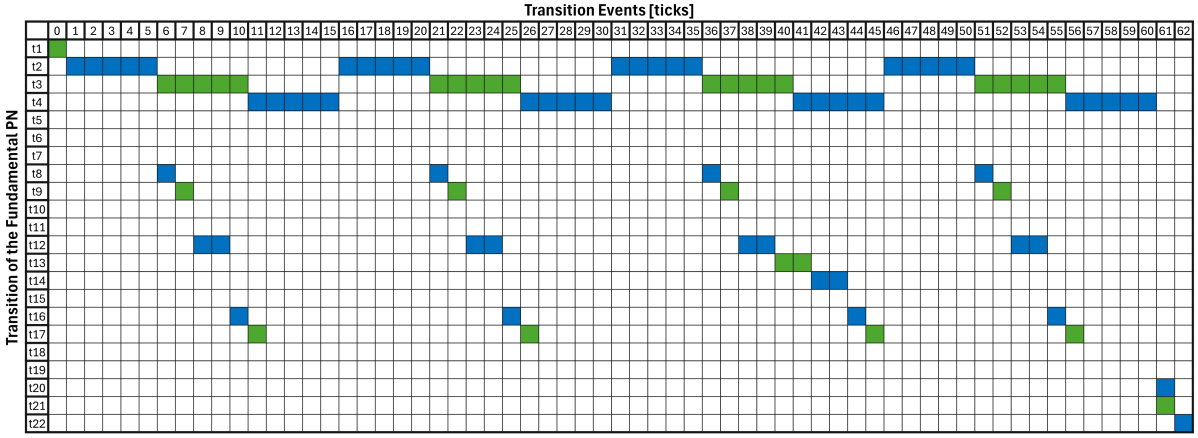
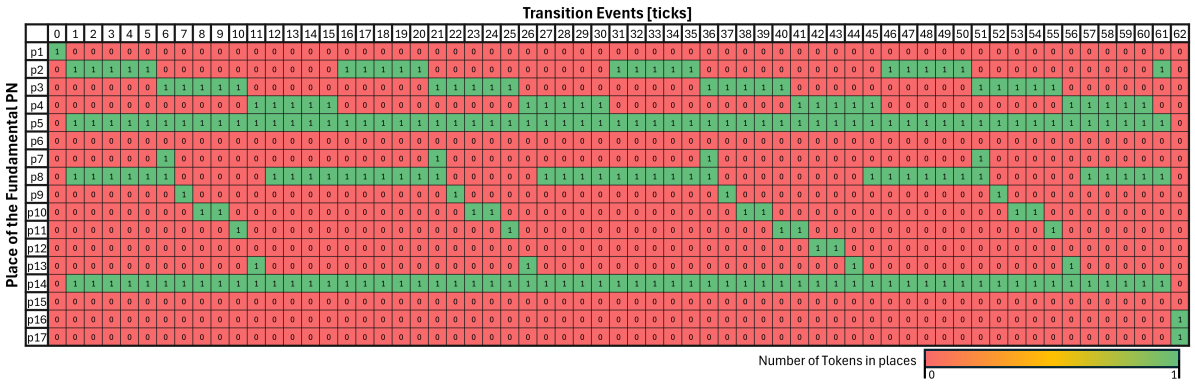


Figure 47 summarizes the evolution of tokens in each place during scenario #1, reflecting the agent's behavior over time. Since the control and FT threads are synchronized, the structural property of repetitiveness is evident, and both threads can execute repeatedly, as shown in the behavior of places $p2$, $p3$, $p4$, $p7$, $p8$, $p10$, and $p11$. Another observed property is conservativeness, where a token is created and destroyed in place $p7$. Moreover, under both regular operation and fault conditions, the fundamental PN remains structurally bounded and safe, as no place contains more than one token simultaneously. In this synchronized scenario, the PN exhibits liveliness and reversibility, as evidenced by the tick 62.

Figure 47 – Evolution of tokens in places of fundamental PN for scenario #1.



Scenario #1 confirms that the fundamental PN operates safely and efficiently under synchronized conditions. The agent demonstrates structural safety, boundedness, and the ability to handle local operations while maintaining synchronization. This scenario validates the robustness of the fundamental PN in managing operations under these conditions.

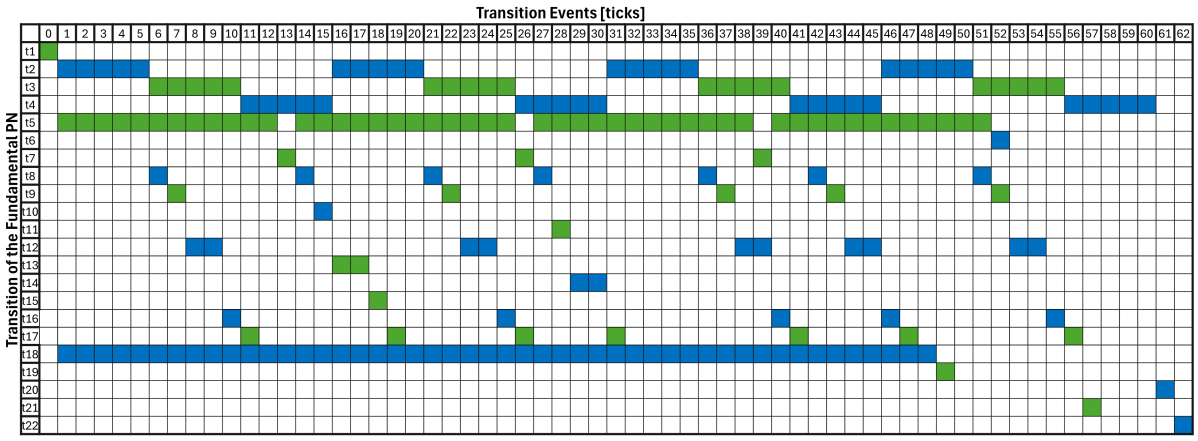
Scenario #2 extends the behavior observed in scenario #1 by incorporating both the activation of the collaboration request thread every 12 ticks and the model improvement process, which occurs after 48 ticks. Despite these additions, synchronization between the control and FT

threads is preserved. In this scenario, the agent does not detect local faults during its operation, however, it responds to external collaboration requests for fault detection, diagnosis, and recovery.

Figure 48 illustrates the evolution of the fundamental PN for scenario #2[5]. As depicted, the agent performs local fault detection cycles between ticks 6–11, 21–26, 36–41, and 51–56, always concluding that no faults are detected. Simultaneously, every 12 ticks, the agent receives a collaboration request within the CF* approach. For instance, at tick 13, the agent receives a collaboration request for fault diagnosis. Since the FT thread is available, transitions $t8$ and $t10$ are triggered, and then activating the transition $t13$ to respond to the diagnosis request. After fulfilling the request, the FT thread is released, as seen between ticks 13 and 18. The same process applies to a fault recovery request between ticks 26 and 31 and another diagnosis request between ticks 42 and 46.

Additionally, scenario #2 includes the model improvement process, triggered after 48 ticks via transition $t18$. As illustrated, this process operates independently, ensuring that it does not interfere with the agent's other operational threads.

Figure 48 – Gantt diagram for performance analysis of the fundamental PN for scenario #2.

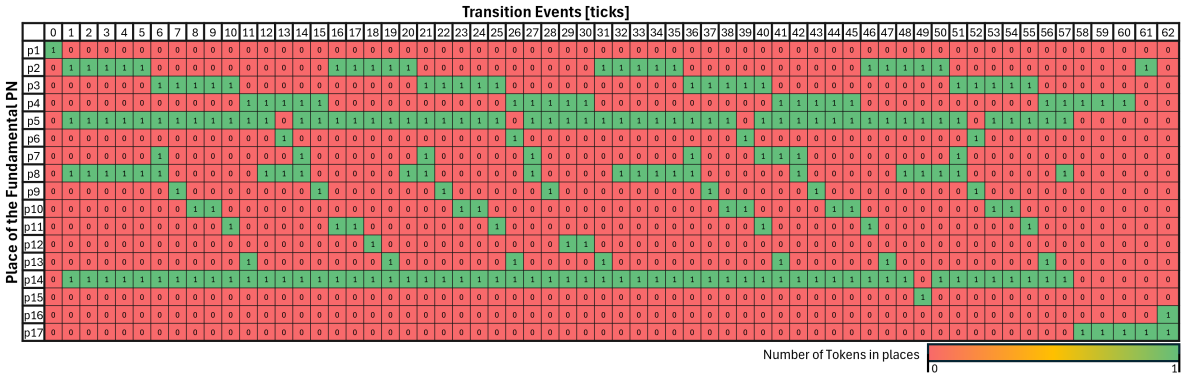


All information regarding token evolution for the agent's behavior in scenario #2 is summarized in Figure 49, which depicts the temporal sequence of tokens throughout the agent's execution. Similar to scenario #1, the execution of control and FT threads remains synchronized, preserving the PN structural and Behavioral properties. This ensures that the control and FT threads can execute repeatedly, as observed in places $p2, p3, p4$ and $p7, p8, p10$, and $p11$. Another key property observed is conservativeness, where a token is created and destroyed in place $p7$. This case occurs when receiving control data (e.g., at tick 6 or tick 21) and when processing a collaboration request (e.g., at tick 14 or tick 27 in place $p7$).

[5] Check the video of PN for scenario #2 on https://youtu.be/V9CV0IK2I0w

Moreover, under the operational conditions of scenario #2, the fundamental PN remains structurally bounded and safe, as no place contains more than one token simultaneously. Additionally, the synchronized operation ensures that the PN exhibits liveness (no deadlocks) and reversibility (returning to the initial state), as confirmed by the agent's state at tick 62.

Figure 49 – Evolution of tokens in places of fundamental PN for scenario #2.



Overall, scenario #2 validates the ability of the fundamental PN to handle additional processes, such as collaboration and model improvement, while preserving its structural safety and synchronization.

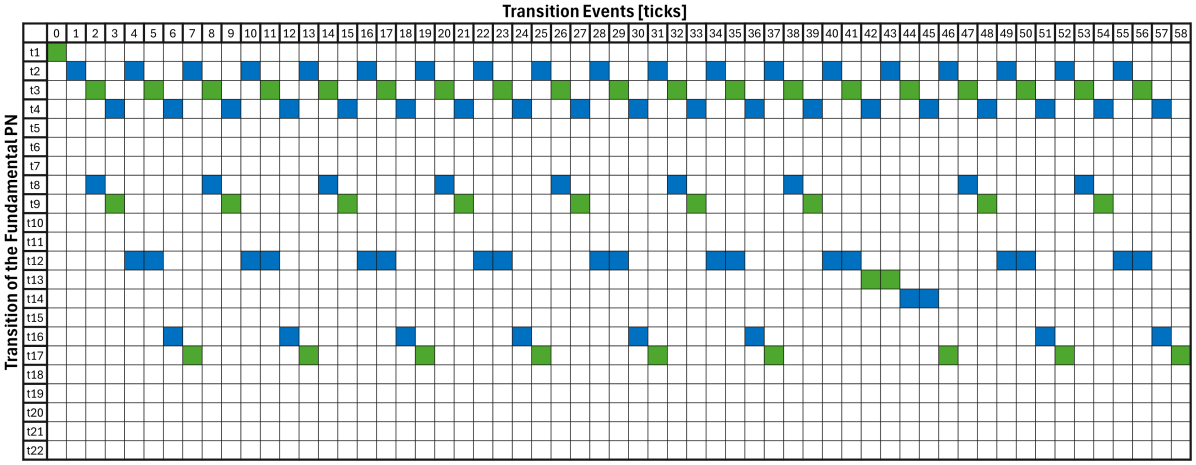
Scenario #3 represents the agent's behavior when the control cycle is faster than the FT cycle. To illustrate the de-synchronization and its impact, collaboration and model updating processes have been omitted in this scenario.

Figure 50 shows the fundamental PN evolution for Scenario #3[6]. The control cycle occurs every 3 ticks, as observed from tick 2 forward, where transitions $t2$, $t3$, and $t4$ are executed. In contrast, the FT thread requires 6 ticks per cycle in the most favorable case, when no fault is present, as can be seen from tick 2 to tick 37. However, when a fault occurs, as illustrated between ticks 38 and 45, the execution of additional transitions ($t13$ for diagnosis and $t14$ for recovery) after fault detection ($t12$) extends the FT cycle to 9 ticks. This further compromises the synchronization between the control and FT threads.

All information regarding token evolution for the agent's behavior in scenario #3 is summarized in Figure 51, which depicts the temporal sequence of tokens throughout the agent's execution. This scenario highlights a phenomenon that significantly impacts FT responsiveness, absent in the previous two scenarios. As the control cycle proceeds quicker than the FT cycle, tokens accumulate in place $p7$ (the data buffer) over the progression of ticks. For instance, at tick 53, there are 10 tokens in this place. This accumulation indicates that the agent may be analyzing

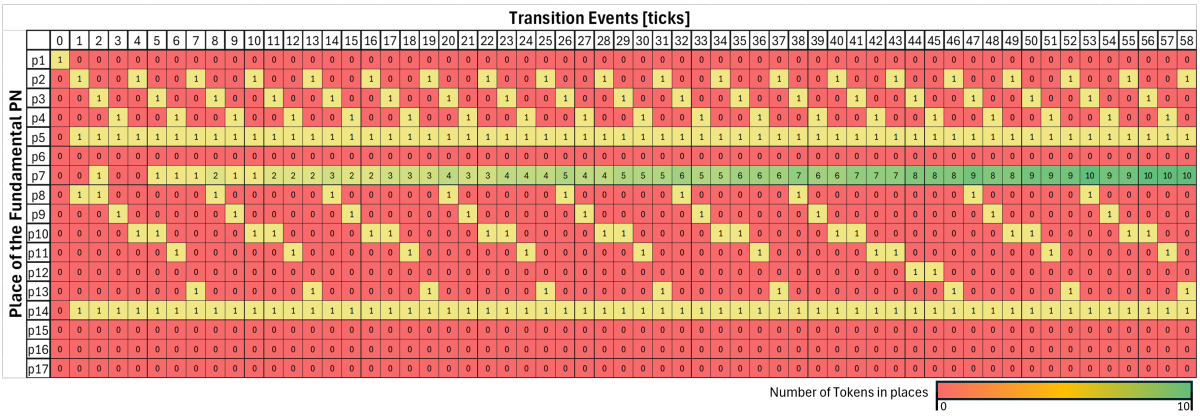
[6] Check the video of PN for scenario #3 on https://youtu.be/7vHaDs2dlWg

Figure 50 – Gantt diagram for performance analysis of the fundamental PN for scenario #3.



outdated data, reducing the efficiency and robustness of its FT operation. Corrective measures should be implemented to mitigate this issue, such as increasing the number of tokens in place $p8$ when the agent's operation starts, resulting in a faster and more responsive FT processing.

Figure 51 – Evolution of tokens in places of fundamental PN for scenario #3.



If collaboration requests were reintroduced under these de-synchronized conditions, culminating in the accumulation of tokens in $p7$ would be exacerbated, potentially leading to system instability. This reinforces the importance of maintaining synchronization between threads to ensure bounded behavior of the fundamental PN.

The three scenarios illustrate the flexibility and robustness of the fundamental PN under varying operational conditions. The first two cases highlight the agent's ability to maintain the properties of safe, bounded, reversible, live, consistent, repetitive, and conservative, while the third case demonstrates the challenges posed by de-synchronization. These examples underscore the importance of system-level control mechanisms to ensure consistent and reliable operation of the CF* approach.

4.5 SUMMARY

This Chapter presented a comprehensive formal modeling and analysis of agent behavior using the Petri nets formalism, ensuring a systematic and structured representation of the CF* approach.

Initially, the agent dynamic PN model with CF* behavior was introduced, referred to as the fundamental PN. This model was developed using timed transitions to represent complex operations, for example, collaborative fault detection, diagnosis, and recovery. The fundamental PN consists of four main threads: control, FT, interaction for collaboration, and model improvement. The timed transitions within these threads can be further exploded to enhance the specificity of the agent's behavioral modeling.

Subsequently, the transitions related to fault detection, diagnosis, and recovery were expanded to provide a more detailed representation of their execution, illustrating local and collaborative behaviors for each stage of FT.

Then, a formal analysis of the fundamental PN was conducted to validate key structural and behavioral properties. This analysis confirmed that the proposed model ensures deadlock-free execution, supports repetitive and reversible operations, and maintains consistency, thereby preserving the agent's ability to control the system, handle faults, collaborate with other agents, and improve its models continually.

Finally, the fundamental PN performance analysis was conducted across three different scenarios, demonstrating the impact of synchronization between control and FT cycles. The analysis findings highlighted that properly synchronizing these cycles is crucial to preventing performance degradation, ensuring timely responses to fault occurrences, and maintaining the agent's overall stability.

The findings in this Chapter provide a formal and validated foundation for implementing CF*-based agents to enable collaborative FT in CPS.

5 EXPERIMENTS AND RESULTS

The I4.0 has driven the digitization based on the adoption of CPS to introduce intelligence and autonomy into industrial systems, enhancing adaptability and efficiency in manufacturing processes. However, the increasing complexity of these systems demands new FT approaches to maintain flexibility and robustness, ensuring operational continuity and reliability in dynamic and decentralized environments.

Chapter 3 introduced the conceptual architecture of CF*, consisting of a meta-model, interaction protocols, detailing the modules responsible for FT and their role in the collaborative operation of autonomous agents contributing to the distributed approaches for improves the robustness, flexibility, and scalability of CPS, enabling cyber-physical components to collaborate in fault detection, diagnosis, and recovery. In Chapter 4, a PN based modeling was employed to formalize the agent's behavior within the CF*, allowing the structural and behavioral analysis of FT stages of CF*.

This Chapter focuses on deploying the proposed FT approach, the CF*, by instantiating the agents in a case study as a proof of concept. The experimental results evaluate the effectiveness of CF* in the fault detection and diagnosis stages and finally establish a conjecture for the recovery stage, which will be explored in depth in future research succeeding this thesis.

The remainder of this Chapter is structured as follows:

- Section 5.1 presents the CPS workbench ARENA and describes its development and agent-based infrastructure, used as a scenario for the proof of concept.
- Section 5.2 describes the experiments and discusses the results comparing the LFT, MCFT and CF* approaches for the fault detection stage.
- Section 5.3 exhibits the utility of the 7CM metric to improve CF* collaboration parameters, based on fault detection experiments.
- Section 5.4 describes the experiments for fault diagnosis, implementing different diagnosis models and conducting statistical tests to validate the findings.
- Section 5.5 discusses how the recovery stage of the CF* could be integrated into the case study explored in the previous Sections, even though the fault recovery stage is not assessed in the results of this thesis.

- Finally, Section 5.6 summarizes the key findings discussed in this Chapter.

5.1 CASE STUDY TO PROOF OF CONCEPT

This Section outlines the case study explored in this thesis to deploy FT in CPS adopting the proposed CF* approach.

One of the challenges in integrating collaborative FT in CPS environments lies in meeting essential requirements such as robustness, flexibility, and scalability. However, scientific and technological progress in this area is hampered by the lack of real-world CPS applications for experimentation, which obstructs the development and validation of FT in practical scenarios. As a result, experimentation in CPS environments is often limited to virtual platforms (TAN *et al.*, 2019) or costly large-scale test environments (CHEN *et al.*, 2017; QUEIROZ *et al.*, 2021). The inability to accurately assess model disturbances, such as implementation errors, bugs, latency, and communication issues, leads to discrepancies between validation results obtained from real and virtual models. Additionally, the absence of real cyber-physical components limits the ability to develop new approaches, formulate hypotheses, and conduct meaningful tests and validations, thereby reducing the practical scientific contribution of research in this field.

To address these challenges, a real CPS environment was developed as a laboratory scale (mock-up) workbench, named ARENA, designed for this case study. ARENA consists of multiple cyber-physical components, enabling the implementation and exploration of CF*. As a laboratory scale, this workbench aims to replicate a real CPS scenario, serving as a benchmark for research and development in FT for CPS. The developed workbench represents a logistics warehouse, where cyber-physical components autonomously manage and execute warehouse operations.

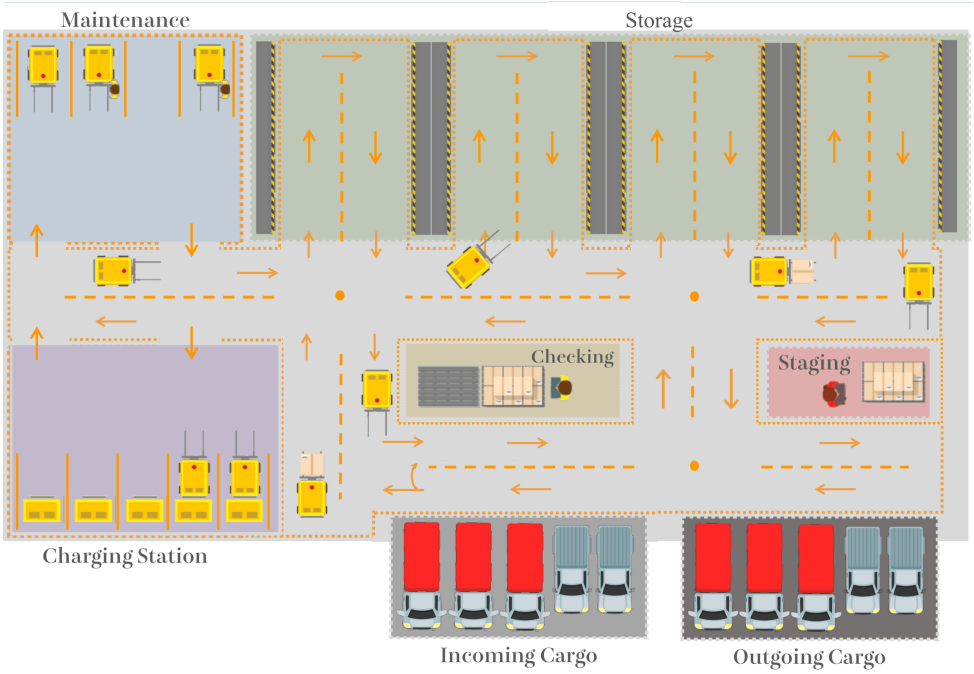
The logistics warehouse is a relevant part of the I4.0 supply chain, representing a dynamic process with several concurrent tasks due to the simultaneous entrance and exit of products (MOUFADDAL *et al.*, 2020). Proper product allocation by storage racks, product transportation by multiple mobile robots, task scheduling, and compliance with product requirements introduce significant complexity, making it a rich environment for research on FT in CPS. Additionally, unforeseen events can dynamically impact system operations and requirements, such as introducing new products, robot failures, battery depletion requiring task reallocation, or storage issues related to rack conditions, such as temperature or humidity deviations. These characteristics make the logistics warehouse an attractive and relevant case study for this thesis.

5.1.1 Case Study Description

The inspiration for ARENA comes from a real logistics warehouse in Brazil that requires a fully automated process. According to Berg and Zijm (1999), the type of logistics warehouse adopted in this work is classified as a “distribution warehouse”. In this typology, logistics warehouses receive products from various suppliers and distribute them to different customers. This model has gained significant attention from large companies and investors due to the rapid growth of e-commerce in the last decade.

Figure 52 illustrates the floor design of the adopted warehouse. The layout includes traffic rules to guide the routes of mobile robots and is divided into seven sectors based on their specific functions.

Figure 52 – Warehouse layout floor reproduced in the case study.

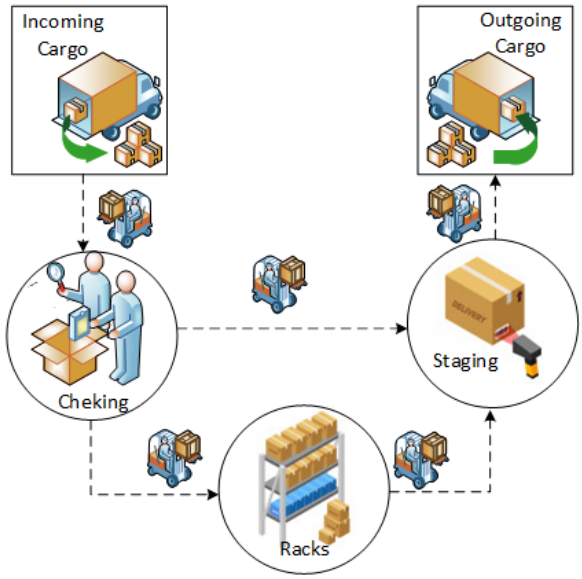


The maintenance sector is responsible for equipment and robot maintenance, while the robot charging station sector provides an area for robots to recharge their batteries. The incoming cargo sector is designated for unloading products upon arrival at the warehouse, followed by the checking sector, where products are inspected to ensure they are in proper condition. The storage sector consists of racks that store products under the required temperature and humidity conditions, with eight aisles providing access to five racks. The staging sector verifies whether products have maintained their integrity while stored in the warehouse, ensuring they have not been damaged. Finally, the outgoing cargo sector is responsible for loading products for dispatch

and exit from the warehouse. This structured layout ensures an organized flow of operations, facilitating the management and automation of warehouse logistics.

In the ARENA warehouse, the product flow, as illustrated in Figure 53, is supported by three fundamental cyber-physical components, namely, products, racks, and robots. Each of these elements plays a crucial role in ensuring the efficiency of warehouse operations.

Figure 53 – Flow of products inside the warehouse for the case study.



In this warehouse, products move from arrival to shipment, passing through several stages, including checking, storage, and staging before shipment. The product integrity and quality are monitored upon arrival, in the checking station, and before departure in the staging station to ensure that products enter and leave the warehouse in perfect condition. The warehouse is predominantly used for e-commerce products, which means it handles small packages with irregular, seasonal, and complex product flows characterized by indeterminate types, sizes, and weights.

Racks serve as storage structures that efficiently organize products, maximizing the use of available space. They can store products that require specific climatic conditions, with controlled temperature and humidity. The strategic arrangement of racks contributes to optimizing time and resources during warehouse operations.

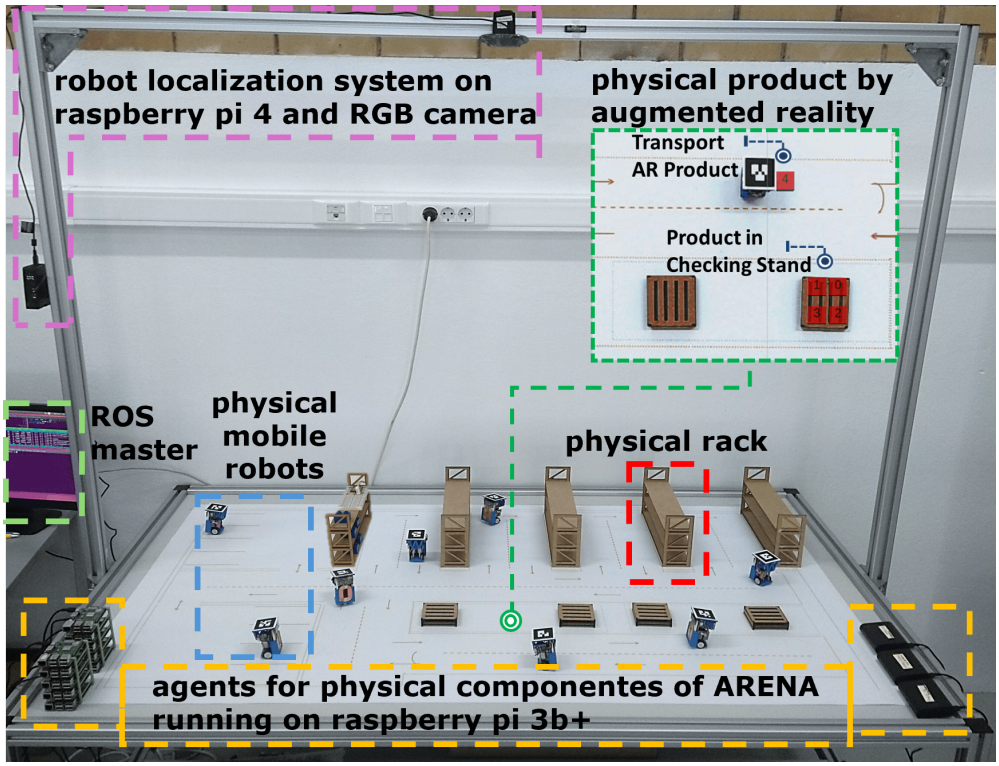
Forklifts, represented in this study by AMR, are responsible for moving products between different sectors of the logistics warehouses. They can load, unload, and transport products from the incoming cargo area to the racks and later from the racks to the staging and shipment areas, streamlining operations and reducing the need for manual intervention. The automation provided by robots enhances the accuracy and speed of operations while minimizing

errors and product damage. Additionally, AMR can move to the charging station or maintenance sector when needed. If robots require recharging, they are directed to the charging station. If maintenance is necessary, they move to the maintenance sector. Furthermore, if product transportation demand is low, idle robots can use this sector as a parking area, reducing traffic complexity within the warehouse.

5.1.2 ARENA Workbench Development and Components

The ARENA workbench contains a structure representing the dimensions of the logistic warehouse on a small scale, providing the similar characteristics and operating areas as a real warehouse. The ARENA comprises table support and physical components, such as small-scale AMRs, racks to store the products, and physical products represented by Augmented Reality (AR), as the robots cannot physically transport real items. The ARENA components can be seen in Figure 54. This structure reproduces the activities and interactions in a small-scale logistic warehouse. The infrastructure gathers the necessary apparatuses to represent the functionalities of the warehouse.

Figure 54 – ARENA workbench representing the case study in laboratory scale.

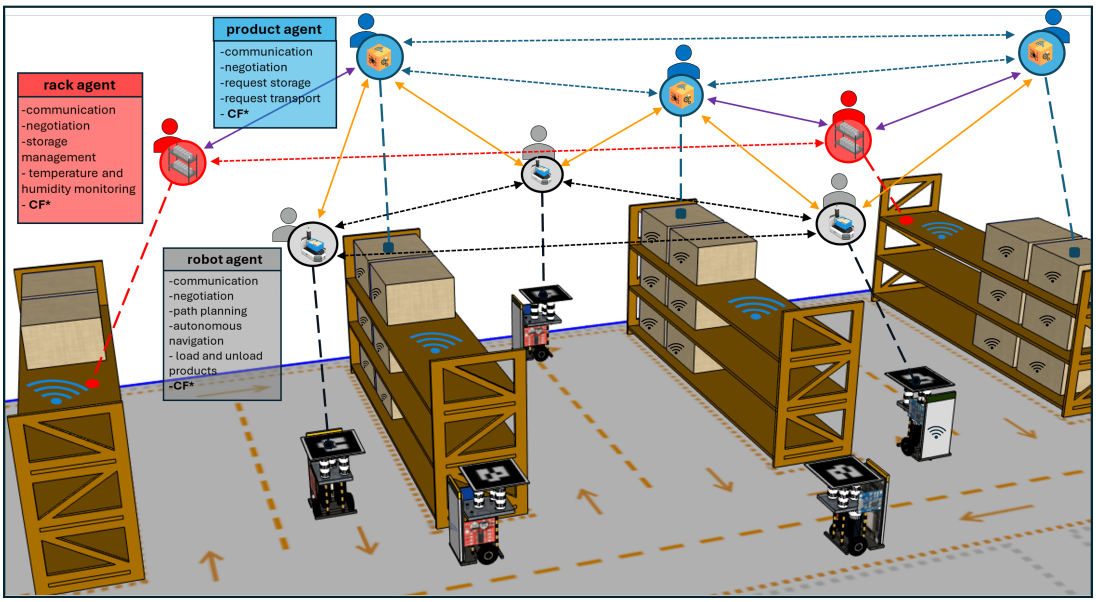


To enable ARENA operation as a CPS, computational devices are assigned to each physical element, i.e., mobile robots, racks, and products, allowing communication, negotiation

and data analysis. Cyber components interact with their physical counterparts through a network, connecting the physical and cyber layers. To ensure a distributed architecture, MAS is used, where each physical component has a dedicated agent running on a processing thread. These agents handle decision-making, control tasks, and implement the CF* approach.

A fully decentralized system was adopted in the case study, avoiding a central node that could introduce processing bottlenecks or single points of faults, which could compromise warehouse operations. Figure 55 illustrates this distributed CPS concept applied to ARENA. Agents within the system communicate with one another, even across different classes (e.g., robots exchanging information with products and racks), enabling autonomous and coordinated warehouse management.

Figure 55 – Overview of the CPS for ARENA case study.

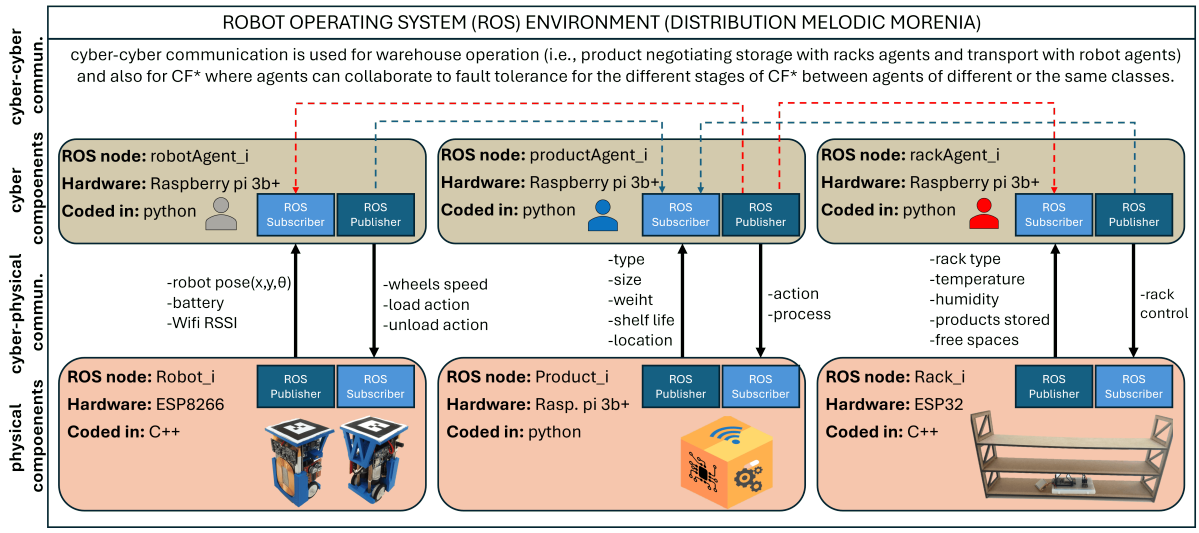


The MAS-based CPS was developed using the Robot Operating System (ROS) Melodic. While ROS is not the most widely used platform for the MAS implementation when compared to alternatives like Java Agent DEvelopment Framework (JADE), it is highly effective for operating mobile robots, which have the most significant complexity due to localization, navigation, control, and path planning requirements. All agents in the cyber layer were developed in Python, a programming language well suited for MAS development and fully compatible with ROS. These agents run on Raspberry Pi 3B+ microcomputers equipped with a 64-bit ARMv8 quad-core processor (1.4GHz) and 1GB RAM, running Lubuntu 18.04.

Integrating physical components and agents follows the ROS publisher/subscriber communication model. Each physical element interacts with its respective agent, as depicted

in the cyber-physical communication diagram (Figure 56). The negotiation process within ARENA, such as assigning a product to a storage rack and selecting the suitable robot for transport, is conducted through agent negotiation, i.e., cyber-to-cyber interaction following a proper negotiation protocol described in the Appendix D.6.

Figure 56 – Description of the ARENA cyber-physical components in ROS environment.



The physical components of mobile robots are equipped with two Direct Current (DC) motors and an ESP8266 microcontroller that digitizes the position, battery voltage, and Received Signal Strength Indication (RSSI) information for their corresponding cyber agents. Each robot agent, running on a Raspberry Pi, due to requires considerable processing power and is responsible for navigation, employing the Dijkstra's algorithm to optimize routes and a Proportional Integral Derivative (PID) controller to regulate the motor speed, ensuring a smooth trajectory tracking and collision avoidance. Depending on the experiment, the ARENA setup supports configurations with two to eight robots. Possible faults in this cyber-physical component include motor malfunctions or localization and navigation errors. According to CF*, each robot agent must implement fault detection, diagnosis, and recovery models with collaborative capabilities to enhance FT.

Each product has an associated cyber agent containing metadata such as type, weight, size (i.e., the number of required storage slots), storage conditions, shelf life, and deadlines. Due to the small scale of ARENA, physical products are represented using AR, allowing robots to simulate transport operations. In real environments, physical products could be integrated with IoT sensors to feed data into the cyber layer. Possible faults include delays in movement or missed deadlines. According to the CF* approach, product agents must incorporate FT models, leveraging collaboration to improve individual fault-handling capabilities.

The physical components of the racks are equipped with an ESP32 microcontroller and a DHT22 sensor to monitor temperature and humidity, ensuring compliance with controlled storage conditions. Each rack contains one hundred storage slots, with each product occupying one or more slots based on its size. In the cyber layer, a rack agent manages storage conditions and executes organization algorithms to optimize the product storage. Key faults scenarios include storage management errors or malfunctions in temperature and humidity sensor systems, with this latter case being the fault explored for proof of concept in this thesis. Fault detection, diagnosis, and recovery mechanisms must be embedded in each rack agent to ensure the reliable FT through collaboration.

Table 13 summarizes the cyber-physical components present in the ARENA, their roles, and features, and indicates the class of agents used in the CPS to operate the warehouse.

Table 13 – Cyber-physical components in the ARENA case study.

Cyeber-Physical Component	Agent Class	Quantity in the ARENA	Role	Features
mobile robot	robot agent	2 to 8	transport products	navigation path planning battery management RSSI management negotiation with products to transport
rack	rack agent	5	store products	space management temperature and humidity monitoring negotiation with product to storage
product	product agent	according to demand	be transported and stored	requirements to be transported requirement to be stored negotiation with robots to transport negotiation with racks to storage

A comprehensive description of ARENA, including its functions, details of the cyber-physical components, agent negotiation, and interaction diagrams, is provided in Appendix D. This appendix also presents experimental results demonstrating the decentralized and distributed capabilities of the ARENA, as implemented in the MAS-based CPS proposed in the (PIARDI *et al.*, 2023).

5.1.3 How the ARENA Case Study was Used for the CF* Proof of Concept

Although the ARENA workbench represents a complete and complex CPS environment for logistic warehouse operations, using its full operational capacity to validate the proposed CF* approach would shift the focus away from the FT approach itself and toward the case study. During the development of this thesis, considerable effort was dedicated in the first years of this work to designing and implementing the ARENA workbench. However, due to the system's inherent complexity and the multiple interdependencies among its components, applying the

CF* approach across the full spectrum of ARENA functionalities would demand significant attention to implementation details, potentially hindering a focused and in-depth analysis of the CF* contributions.

To ensure that the emphasis remains on the proposed CF* approach, a specific case study was designed using the ARENA workbench to isolate and evaluate the different stages of the FT. The experiments were structured to focus individually on the fault detection (with experiments for the 7CM metrics), fault diagnosis, and fault recovery stages, as defined in the CF* approach. Each stage was analyzed within a controlled context using selected ARENA components, thereby enabling a systematic exploration of the capabilities and benefits of CF*.

The fault detection stage was conducted using data from temperature and humidity sensors embedded in the physical layer of the rack components. This stage includes performance comparison for fault detection in the sensor between the CF* approach and two traditional baseline approaches, namely the LFT and MCFT, deploying the detection model in five rack agents.

The fault diagnosis uses the ARENA rack sensor by injecting multiple fault scenarios and instantiating CF* with five different diagnosis models among twelve rack agents. This fault injection allowed the assessment of the collaborative capabilities of CF* identifying root causes of faults. A statistical analysis was conducted to validate the performance improvements achieved by CF* when compared to LFT approach.

Finally, while the fault recovery stage was not empirically evaluated in the same depth, a conjecture outlines how CF* supports recovery actions for the case explored in the detection and diagnosis stages. A more comprehensive and quantitative investigation of the fault recovery stage is left as future work beyond the scope of this thesis.

5.2 EXPERIMENTS FOR FAULT DETECTION IN THE RACK SENSOR

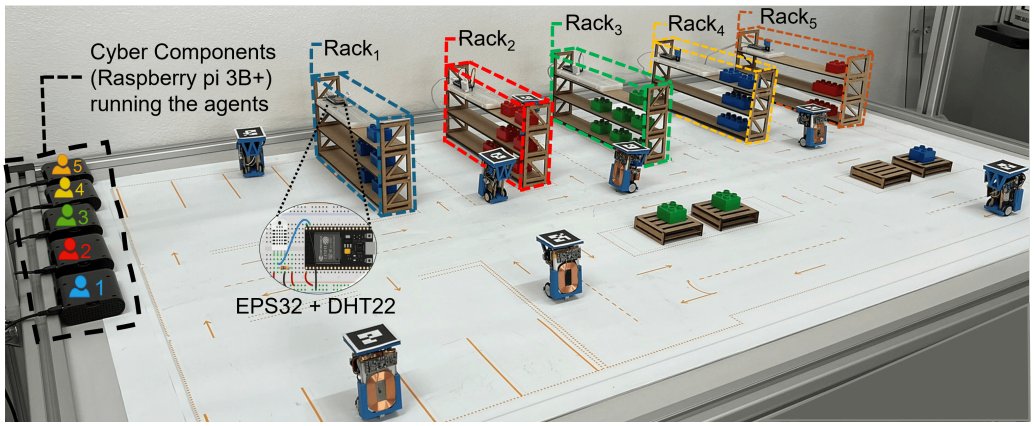
The experiments for fault detection, compare the performance of CF* approach with the LFT and MCFT. The results of this section are published in (PIARDI *et al.*, 2025b).

The experiments for fault detection focuses on detect fault in the rack condition, with temperatures ranging between 17 and 20 $^{\circ}C$ and the air humidity between 35% and 40%. Deviation from the expected sensor output in terms of temperature and humidity can compromise the storage quality and integrity of the stored product. Therefore, it is essential to ensure the fault detection efficiency in terms of temperature and humidity anomalies, paving the way for the next

FT stages, namely diagnosis and recovery, and minimizing the fault impact on storage quality.

Figure 57 shows the ARENA setup for the detection experiment with five cyber-physical components of the rack class. Each rack's physical part comprises a DHT22 sensor connected to an ESP32 microcontroller to acquire the temperature and humidity data and communicate this information via Wi-Fi to the respective agent.

Figure 57 – Five cyber-physical components racks used in the detection stage experiments.



5.2.1 Dataset Elaboration for Fault Detection Experiments

Seven healthy DHT22 sensors were used to acquire a dataset containing seven samples with 1440 temperature and humidity observations for the experiments, five of which were used for training the detection models and two for testing. Each observation was obtained within a 10-second time frame, totaling 4 hours of data acquisition. The expected output of healthy sensors is expressed in Equation (16) as follows:

$$s(t) = h(t) + \eta, \quad \eta \sim N(0,\delta_\eta^2) \tag{16}$$

where $h(t)$ and $s(t)$ is the (temperature or humidity) sensor output without noise or fault and sensor data output respectively, expected in time t of healthy sensors and $\eta \sim N(0,\delta_\eta^2)$ is the noise with zero mean and low variance associated with each sensor.

Among the seven sensors, one is distinct because it is not subjected to fault injection, i.e., a fault-free data sample, to verify the ability of the detection models to avoid false positives when no faults are present. Meanwhile, the remaining six samples were processed, injecting precision degradation fault into the sensors' output. As described in (JAN *et al.*, 2021), this type of sensor fault occurs because of sparse connections in the circuit, noise resulting from

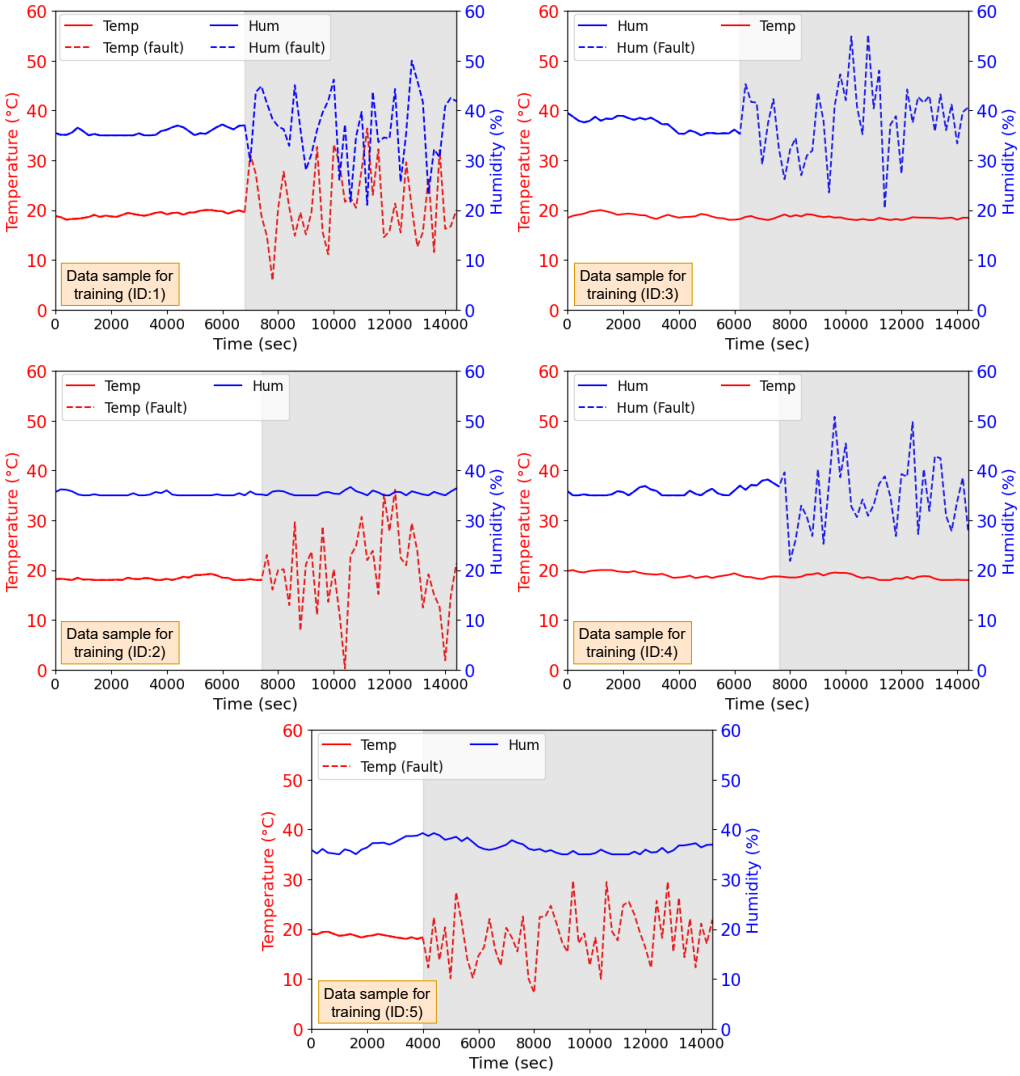
high-frequency operations, and physical damage to the sensor. When a fault manifests itself, a precision degradation is injected according to the model described in Equation (17).

$$s_{precision}(t) = h(t) + \eta + \nu, \nu \sim N(0,\delta_{\nu}^2) : \delta_{\nu}^2 \gg \delta_{\eta}^2 \tag{17}$$

The noise associated with a precision fault has zero mean and high variance. Faults can manifest individually in temperature or humidity or concurrently affect both measurements.

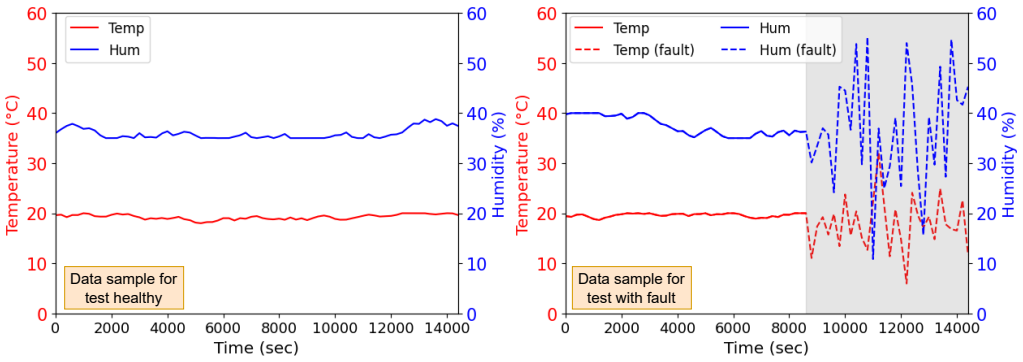
For the detection experiments, the dataset was divided into two parts. The first part contains five samples with precision faults injected, as shown in Figure 58. These five samples were used to train the fault detection models embedded in the respective agents responsible for the racks according to component ID, e.g., $agent_1$ responsible for $rack_1$ was trained with the dataset sample with ID 1.

Figure 58 – Dataset used for training the fault detection models for each agent.



The second part of the dataset (shown in Figure 59) contains two samples used in different experiments. The sample on the left shows the data obtained from a healthy sensor to represent a fault-free experimental scenario. The sample on the right contains data obtained from the sensor with precision degradation, representing an experimental scenario with fault manifestation.

Figure 59 – Dataset with fault-free and faulty samples used in fault detection experiments.



5.2.2 Experimental Procedure

To compare the fault detection capabilities of traditional approaches, i.e., LFT and MCFT as presented in Chapter 2.3.2, with the collaborative fault detection of the proposed CF*, a strategy involving a binary classifier based on a SVM algorithm was employed. To achieve this, it is necessary to train the fault detection models (offline process) and deploy them into agents to perform the online fault detection. It is worth noting that the objective of the experiments was not to optimize the classifier model for fault detection (for this, it is possible to consult (YOO, 2020; YANG *et al.*, 2023; CHEN *et al.*, 2011)), but to compare different approaches. Consequently, the compared approaches did not incorporate feature extraction techniques or principal component analysis as a means to optimize the classification models.

The Python library scikit-learn[1] was employed to implement the SVM algorithm (PEDREGOSA *et al.*, 2011), within the rack agents to classify the faults in the rack sensors. The input consisted of a sliding window encompassing sixty temperature and humidity observation each. The output domain was binary, with “0” representing healthy data and “1” representing faulty data with correspondent confidence level for each output (calibrated with Plat Scaling method). All SVM classifier models were trained using the parameters listed in Table 14.

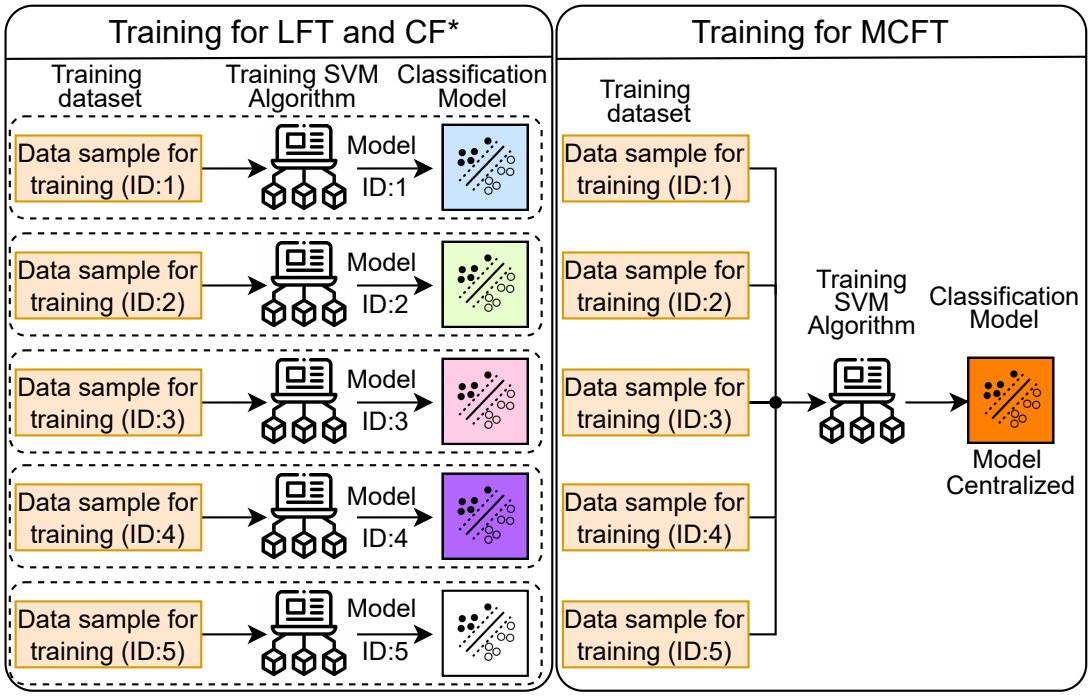
[1] https://scikit-learn.org/

Table 14 – Parameters used in the SVM model for the fault detection experiment.

Parameter	Value or Method
Kernel	rbf
Regularization parameter (C)	1.0
Gamma	scale
Decision function shape	ovr
Probability	true
Probability calibration	Platt scaling
Tolerance for stopping criterion	0.001

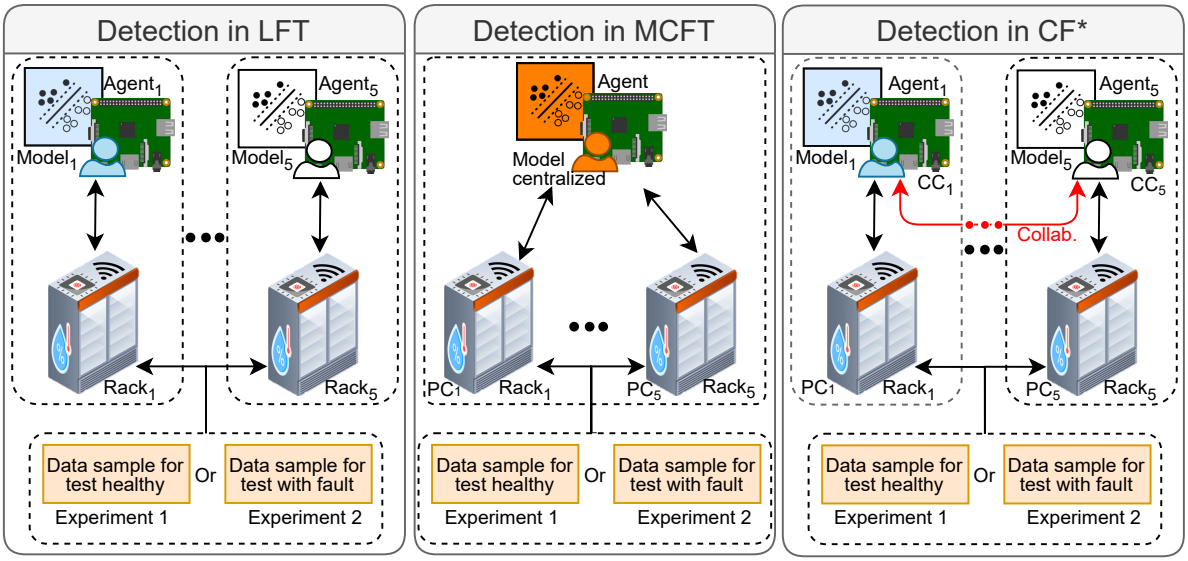
Figure 60 summarizes the classifier training procedure and illustrates the process through which each model was derived. A uniform training procedure was employed for both the LFT and CF*, given that the classifier models were locally integrated within each agent. In contrast, the MCFT consolidates all training data samples to develop a model with enhanced generalization capacity and minimized overfitting. This is expected to yield a superior performance, which is typical of a centralized model trained on a larger dataset.

Figure 60 – Procedure used for training the SVM models for detection experiments.



The experimental procedure for comparing the different FT approaches is shown in Figure 61. Two experiments were conducted for each approach based on the data test from the created dataset (Figure 59). For fault detection using the LFT approach, each model was embedded in the respective rack agents that received temperature and humidity data from the physical part, that is, the respective racks. In the MCFT approach, one agent is responsible for detecting faults in the five racks, and must internally manage the data received from each component and manage the fault detection for each physical device.

Figure 61 – Overview of the experimental procedure developed for fault detection experiments.



For the CF* experiment, each agent was responsible for a physical rack component and performed a self-fault detection and collaboration when needed, according to the collaboration parameters τ, σ, and ρ (as described in Chapter 3.2.2). These parameters were uniformly implemented across all agents for fault detection in CF* as presented in Table 15. The τ parameter was determined via the confidence level of the model's inferences for each observation in the data test. Collaboration was initiated if an inference classified an observation with a confidence level lower than 85% (a threshold established based on experimental practice) for both healthy and faulty data test experiment. The agent selection for collaboration, σ parameter, encompasses all five agent racks of the system since they are compatible. Finally, the decision makes ρ parameters is based on the majority of agent votes.

Table 15 – Instantiation strategies for the collaboration parameters of CF* in fault detection experiments.

CF* Parameter	Instantiation Strategy Adopted	Value
Trigger (τ)	uncertainty or inconclusiveness in SVM algorithm inference	confidence <85%
Selection (σ)	compatibility	collaborate with all rack agents available.
Decision-making (ρ)	most voted	fault detected or fault-free based on majority of votes

5.2.3 Results and Discussion for Detection Stage

In the fault-free experiment, the data sample for testing the healthy sensor condition (see Figure 59) represents a scenario in which the agents are not expected to detect any faults. However, owing to the inherent limitations of the SVM trained models, some observations were

incorrectly classified as faulty.

Figure 62(a) shows the performance of the MCFT, LFT, and CF* approaches to elucidate on the misidentified faults (shaded area in the chart). With a detection model trained on a more comprehensive dataset, the MCFT approach outperformed the LFT approach for all agents. However, in the LFT approach, each agent demonstrated unique fault detection characteristics. Through the collaboration facilitated by the CF* approach, the agents improved the performance of the LFT, reducing the number of incorrect classifications. For example, for $agent_3$, in Figure 62(a) it is possible to verify a significant reduction in misidentified faults (shaded area in the chart) when comparing LFT and CF*.

Figure 62(b) complements the comparison by presenting the observations exposed through a confusion matrix for each approach, showing the accuracy improvement for the fault-free experiment when considering the proposed CF* approach for collaborative detection implemented in each agent compared with the LFT approach, representing an improvement of 13% in F1-score on average.

Table 16 showcases the precision, recall, and F1-score metrics[2] for the fault-free experiment for fault detection stage. The precision score measure the agent's ability to not label a faulty sample as healthy sample. On the other hand, the recall value measures the agent ability to classify healthy samples correctly.The F1-score is the weighted harmonic mean of precision and recall, incorporating the importance of both precision and recall. For each of these metrics, the value of 100 denotes the ideal performance, whereas the value of 0 indicates the least favorable outcome.

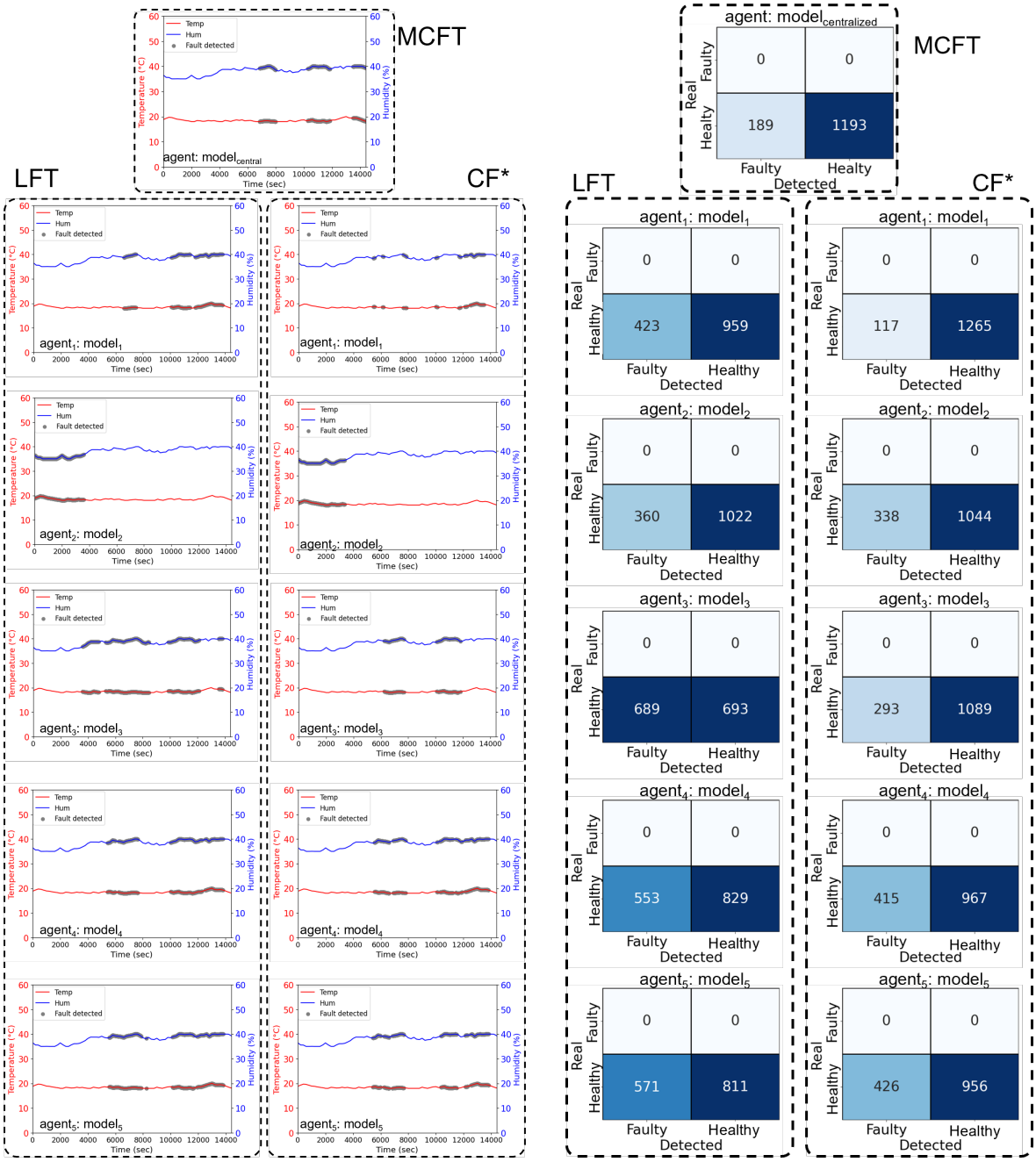
Table 16 – Accuracy for fault detection experiment with fault-free data test.

	-	agent 1		agent 2		agent 3		agent 4		agent 5	
	MCFT	LFT	CF*	LFT	CF*	LFT	CF*	LFT	CF*	LFT	CF*
Precision	100	100	100	100	100	100	100	100	100	100	100
Recall	86	69	92	74	76	50	79	60	70	59	69
F1-score	93	82	96	85	86	67	88	75	82	74	81

Considering the focus on healthy data exclusively, the models' precision metric attains the max score, as no faulty data is present. Consequently, the emphasis shifts to evaluating the recall and F1-score metrics. The MCFT approach exhibits satisfactory fault detection results, with an F1-score of 93%, outperforming all LFT agents, with some agents displaying a low F1-score value, notably $agent_3$, $agent_4$, and $agent_5$ with values of 67%, 75%, and 74% respectively. However, when adopting the agent's collaboration for fault detection, the recall metrics have

[2] The calculations to obtain these metrics can be checked in: https://tinyurl.com/sckit-Metrics

Figure 62 – Comparison between the MCFT, LFT, and CF* for fault detection experiment with fault-free samples.



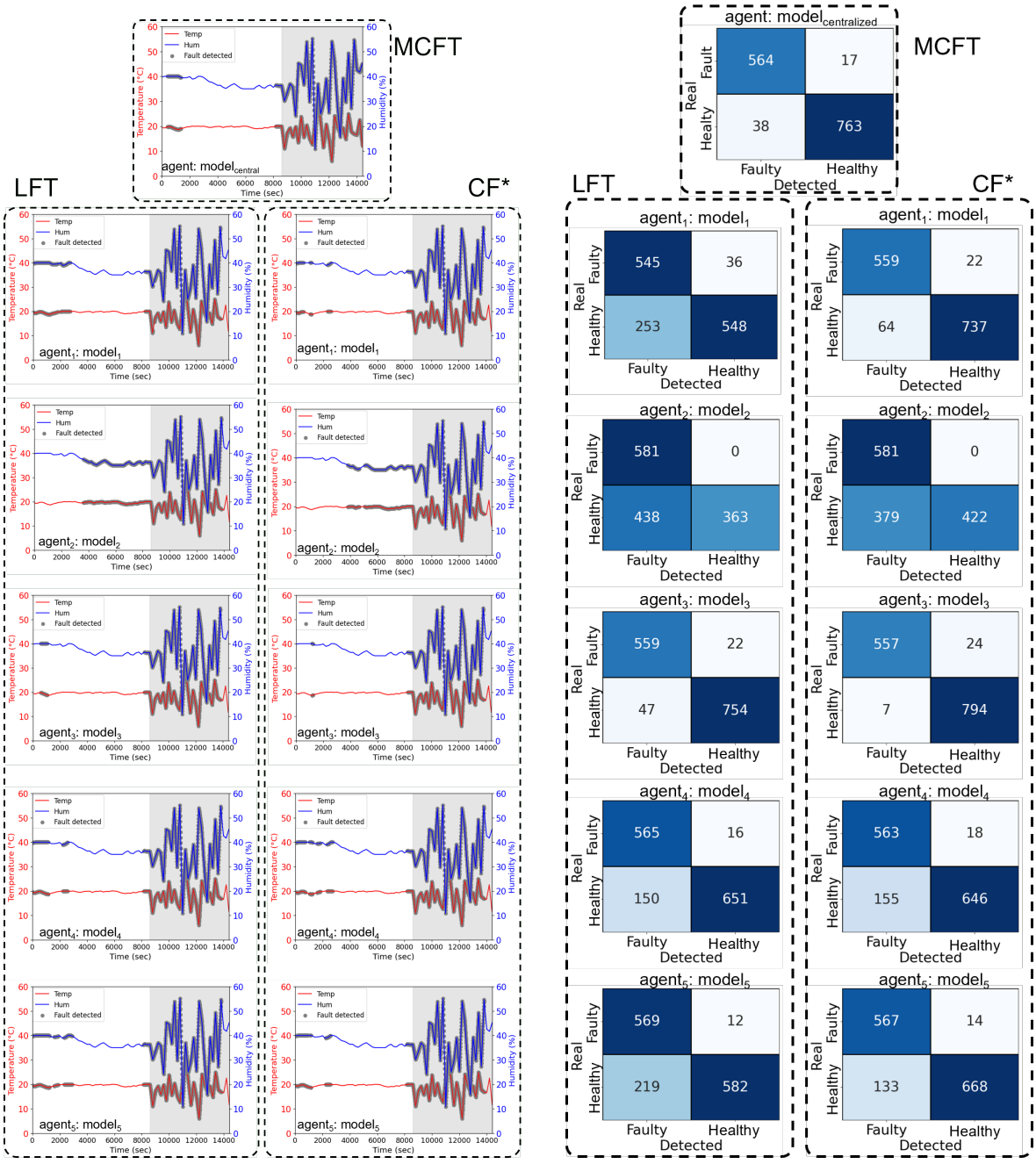
(a) Detection in fault-free data test. **(b) Confusion matrix.**

a discernible enhancement and, consequently, the improvement of F1-score across all agents. Although $agent_2$, the best agent without collaboration, does not maintain its lead in the CF* approach, but it plays a pivotal role in collaboration, significantly bolstering the performance of other agents. Notably, $agent_1$ excels under CF*, surpassing the baseline set by the MCFT by 3.2%. Comparing the MCFT and CF* approaches, it is observed that, on average, the F1-score is higher for MCFT (93% vs. 86%). However, CF* demonstrates competitiveness due to its

distributed approach, which enhances the scalability and avoids the risk of a single failure point.

The second experiment for fault detection, which considered data test with precision degradation faults in the sensors, also demonstrated improved performance in agents that utilized the collaboration protocol in the proposed CF* fault detection approach. Figure 63(a) shows the performance of the MCFT, LFT, and CF* approaches.

Figure 63 – Comparison between the MCFT, LFT, and CF* for fault detection experiment with faulty samples.



(a) Precision degradation fault data test. **(b) Confusion matrix.**

Figure 63(b) shows the observations delineated through a confusion matrix for the

approaches comparison, providing an overview of the performance for the precision fault experiment. The improvement introduced by the collaborative fault detection implemented in the agents was considerable. However, for $agent_4$, the collaboration resulted in a minor induction of error (inserting two falsely healthy and five falsely faulty observations).

Table 17 provides a detailed breakdown of the precision, recall, and F1-score metrics derived from the fault detection experiments with precision degradation fault injection in the data test. Upon further comparison between the LFT and CF* approaches, an enhancement of the precision and recall was observed for the latter, resulting in an improvement of 7% for the F1-score on average. Notably, $agent_3$ exhibited a F1-score improvement of 1% under the proposed CF* approach as compared to the MCFT.

Table 17 – Accuracy for fault detection experiment with precision degradation fault data test.

	-	agent 1		agent 2		agent 3		agent 4		agent 5	
	MCFT	LFT	CF*	LFT	CF*	LFT	CF*	LFT	CF*	LFT	CF*
Precision	98	94	97	100	100	97	97	98	97	98	98
Recall	95	68	92	45	53	94	99	81	81	73	83
F1-score	97	79	94	62	69	96	98	89	88	83	90

5.3 ADOPTING THE 7CM TO ENHANCE THE CF* PARAMETERS

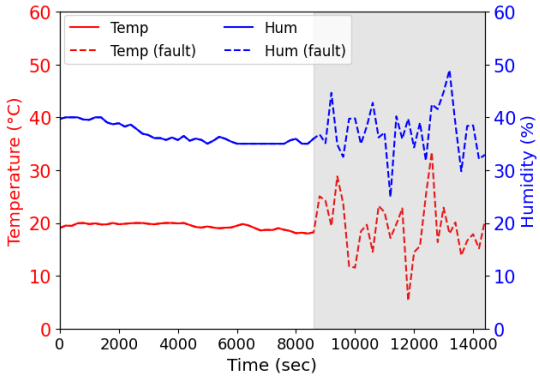
In this Section, the impact of agent collaboration for FT is evaluated using the 7CM metrics (as presented in Chapter 3.6). A similar approach to the one described in the previous Section was followed, employing the same ARENA setup with the same five agents trained using the dataset shown in Figure 58 and the SVM algorithm for fault detection models. However, a different test dataset was used. Although it contains precision fault injections with the same behavior described in Equation (17), it represents a new sample. This evaluation focuses on analyzing how the proposed metrics can indicate potential improvements in the CF* approach. This choice allows the evaluation of the generalization capability of the proposed approach when applied to different test data. The findings presented in this Section have been published in (PIARDI *et al.*, 2024b).

5.3.1 Data Test Used for Experiments

Figure 64 presents the new sample used for the data test elaborated to explore the 7CM. This data test comprises 1440 temperature and humidity observations collected at 10-second intervals, totaling four hours of data acquisition. Within this new data test, the initial

859 observations were deemed healthy, followed by the injection of precision faults into the remaining 581 samples using Gaussian noise in temperature and humidity as presented in the Equation (17).

Figure 64 – Data test used in each agent for the three experiments related to 7CM.



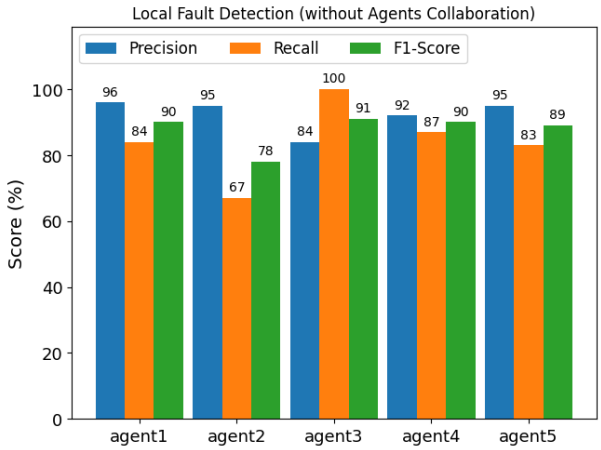
5.3.2 Experiments Evaluation and Results Adopting 7CM

For consistency, the data test was injected into all agents during the three proposed experiments for this Section. In the first experiment, each agent operated independently, detecting faults using only its local model without collaboration. This experiment served as the baseline to evaluate individual performance. In the second experiment, collaboration among agents for fault detection was enabled according to the CF* approach, using identical predefined collaboration parameters (τ, σ, ρ) for all agents. The objective was to analyze the general benefits of collaboration without parameters adjustment. Subsequently, the proposed 7CM evaluation metric was applied to the results obtained from the second experiment to assess the collaborative behavior of each agent. Based on the analysis provided by the 7CM metrics, adjustments were proposed to the collaboration parameters individually for each agent. For the third experiment, the agents collaborated again, but now using the refined collaboration parameters suggested by the 7CM evaluation. This procedure allowed for assessing the impact of tailoring collaboration settings to each agent's specific performance characteristics. The individual performance of the agents across the three experiments was compared in terms of precision, recall, and F1-score for fault detection stage.

Figure 65 displays the individual performance of the five agents with the LFT, i.e., the non-collaborative approach. Individually, a reasonable performance is observed, with the F1-score of $agent_1$, $agent_3$, $agent_4$, and $agent_5$ around 90%, despite $agent_2$ having low recall

values directly impacting its corresponding F1-score. The $agent_3$ successfully identified all fault-free observations, with a recall of 100%, but misclassified 16% of faulty observations as fault-free, resulting in a precision of 84%. In contrast, $agent_1$, $agent_2$, $agent_4$, and $agent_5$ had precision values higher than recall. Therefore, the collaboration is expected to foster collective intelligence, enhancing recall and precision, surpassing the individual capacity of each agent.

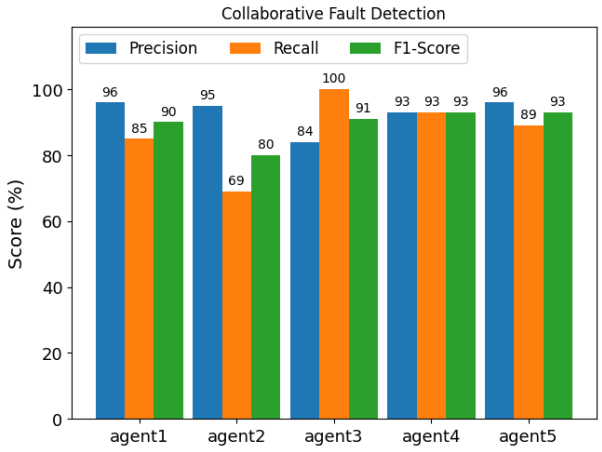
Figure 65 – Individual agents performance for fault detection based on data test for 7CM experiment.



In the second experiment, the collaboration among the agents for fault detection was introduced. For this purpose, exploratory and equal collaboration parameters were assigned to all agents, namely the collaboration trigger $\tau = 0.7$, meaning that a given agent will initiate collaboration if the detection inference result has confidence below 70%. The selection parameter σ selects all agents in the case study, as there are only five agents, and all are compatible. Lastly, the decision-making parameter ρ is configured to classify the sample following the majority of analyses of all agents, i.e., the majority voting rule. Figure 66 presents the performance of each agent in classifying the data test, showing a noticeable improvement in some agent's performance. However, despite the collaboration, $agent_1$ and $agent_3$ maintained their fault detection performances almost stagnant. Nevertheless, they were fundamental in the improvements achieved by $agent_2$, $agent_4$, and $agent_5$, with their F1-score improving by 2.5%, 3.3%, and 4.5%, respectively. It is worth highlighting the considerable improvement in recall for $agent_4$ and $agent_5$ by 6.9% and 7.2%, respectively.

Despite the observed improvement with the introduction of collaboration among the agents, this behavior needs to be assessed to identify aspects that each agent needs to improve regarding the collaboration parameters or detection model. Table 18 presents the results of applying the proposed 7CM (as described in Section 3.6), highlighting the collaboration outputs. The CM_1 metric indicates that few collaboration events occurred, for example, $agent_3$ requested

Figure 66 – Performance of agents with collaborative fault detection before the adjust by 7CM.



collaboration in approximately 1% of its observations to detect faults, while $agent_2$ requested collaboration around 13% of the data test sample. Therefore, it is necessary to appropriately increase the collaboration trigger so that agents can request more collaboration to enhance fault detection and improve the agent's performance.

Table 18 – Result of 7CM for experiment with collaboration for fault detection.

		$agent_1$	$agent_2$	$agent_3$	$agent_4$	$agent_5$
collab. output	qi	1382	1382	1382	1382	1382
	qc	91	179	12	110	78
	ri	26	3	0	2	0
	rc	26	150	7	22	19
	mi	17	6	0	2	1
	mc	22	20	5	84	58
collab. metrics[%]	CM_1	0.07	0.13	0.01	0.08	0.06
	CM_2	0.29	0.02	0.00	0.02	0.00
	CM_3	0.29	0.84	0.58	0.20	0.24
	CM_4	0.19	0.03	0.00	0.02	0.01
	CM_5	0.24	0.11	0.42	0.76	0.74
	CM_6	0.47	0.05	0.00	0.04	0.01
	CM_7	0.53	0.95	1.00	0.96	0.99

Considering the Table 18, through the individual analysis, $agent_1$ requested collaboration in 7% of its observations (CM_1) but failed to leverage its performance with collaborations, as indicated by CM_6, which represents the undesirable collaboration index, being very close to the desirable collaboration index CM_7. This high CM_6 value is attributed to the collective error CM_2 (individually detect fault wrong, and when collaborating, remains wrong) and the collaboration inducing to error CM_4 (i.e., individually detect fault correctly, but when collaborating, is induced to error wrongly). The most desirable behavior when collaborating is correcting a wrong detection (CM_5), which accounts for 24% of collaborations performed for $agent_1$. To improve the collaborative initiative of $agent_1$, its collaboration trigger will be raised to $\tau = 0.85$.

The selection parameter σ remains the same, selecting all agents. Lastly, the decision-making parameter ρ also remains unchanged, considering the majority of votes.

The $agent_2$ stood out as the most proactive entity in initiating collaborations, as shown by the CM_1 value. However, 84% of these collaborations merely confirmed that its local detection decision was correct, as indicated by CM_3. Although not a negative indicator (as these data could be used to retrain its model), it does not directly impact the classification performance, remaining similar to that obtained in the first experiment without collaboration. With this in mind, the collaboration parameters for $agent_2$ were adjusted, increasing the collaboration trigger value to $\tau = 0.9$. The selection parameter remains unchanged, but the decision-making parameter ρ is altered to adopt the decision of the voter with the highest confidence score.

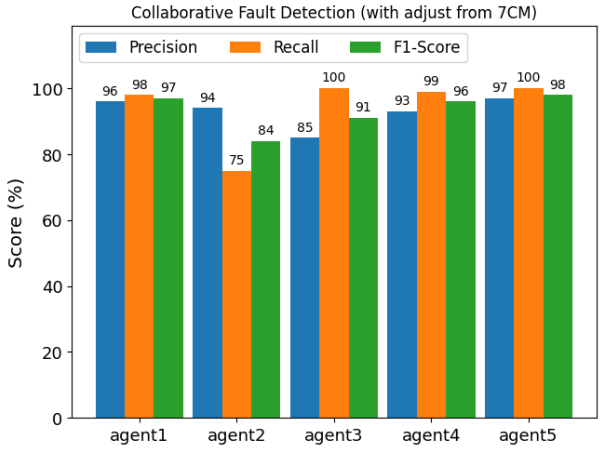
The $agent_3$ and $agent_5$ were the least proactive in initiating the collaboration, with CM_1 at 0.01 and 0.06, respectively. Due to this fact, the obtained corrections (CM_5) did not affect their performance or could have been even better, given that $agent_5$ showed significant improvement when collaborating. These two agents' desired collaborations, expressed by CM_7, are the highest. Therefore, the only change for these agents is to considerably increase the collaboration trigger to $\tau = 0.9$ while keeping the other parameters unchanged.

The $agent_4$ exhibited good behavior, being the second agent to initiate collaboration most frequently, as indicated by CM_1, and showed a high degree of correction, with CM_5 at 76%. This suggests that the collaboration parameter of decision-making ρ is suitable for it. In order to further encourage its collaboration initiative, only the collaboration trigger will be adjusted to $\tau = 0.85$.

Figure 67 illustrates the performance of each agent with the changes in parameters τ, σ, and ρ obtained after analysis and adjustment based on Table 18. The $agent_1$, $agent_4$, and $agent_5$ showed good performance, significantly improving compared to the previous experiments. For example, the recall of $agent_1$ improved by 16.6% compared to the experiment without collaboration and 15.3% compared to the experiment with the agents collaboration. The $agent_3$ showed a slight improvement of 1.2% in precision compared to the experiment with agents collaboration. The $agent_2$ improved the recall by 8.6% and the F1-score by 5% compared to the experiment with agents collaboration but presented a precision decrease of 1%, resulting from incorrect modifications (CM_4) due to the change in the decision-making ρ. Nevertheless, this adopted decision-making parameter positively impacted the recall value and, consequently, on the F1-score, improving by 8% and 5% compared to the local and collaborative experiments,

respectively.

Figure 67 – Performance of agents after the adjustment of the collaborations parameters by 7CM analysis.



5.3.3 Discussion related to the 7CM

The experimental tests conducted in this Section demonstrate the effectiveness of the 7CM metrics in assessing the impact of collaborative parameters in the detection stage of CF*. However, beyond their role in evaluating the agent collaboration behavior and adjusting the collaboration parameters, these metrics offer significant potential for improving the CF* approach.

One promising direction is the integration of 7CM into the online operation of each agent, enabling the continuous performance monitoring and dynamic adjustment of collaboration parameters. This capability allows agents to self-regulate their participation in the collaborative process, adapting to changing conditions and enhancing overall system robustness. For instance, if an agent excessively relies on collaboration, resulting in simply reinforcing its local decision, this behavior may indicate uncertainty or excessive insecurity. In such cases, the agent can adjust its trigger parameters to reduce excessive collaboration requests, preventing both local and other agents processing and bandwidth overload.

Another key aspect is the use of 7CM as a guide for updating the FT models. These metrics can provide insights related to when model adjustments are required. For example, if the evaluation of the 7CM consistently reveals a misalignment between local and collaborative decisions, this discrepancy may indicate the need for retraining with additional data or fine-tuning model parameters to correct or enhance local decision-making.

Finally, while this Section focuses on the application of 7CM in the fault detection

stage, its principles can be extended to the diagnosis stage of CF* by changing the classification between two classes (with a fault or without fault for detection) to a scenario with a list of faults of different types.

5.4 EXPERIMENTS FOR FAULT DIAGNOSIS IN THE RACK SENSOR

To experimentally validate the benefits of collaboration in FT, specifically in the diagnosis stage, a second set of experiments was conducted in which CF* was based on data-driven methods, with agents instantiated using machine learning models for fault diagnosis. This Section describes the case study, dataset elaboration, experimental procedure, proposed evaluation methodology with the deployment of CF* in the agents, the experimental results, and statistical test to validate the proposed approach. The results of this Section are published in (PIARDI *et al.*, 2025a).

The collaborative fault diagnosis was applied in ARENA as previously shown in Figure 57, and described in the fault detection stage (Section 5.2). The study follows the same case study proposal, focusing on rack agents that monitor temperature and humidity to ensure the required storage conditions of products. However, some modifications were introduced to extend the analysis for the fault diagnosis stage of the CF* approach.

First, the case study was expanded from five rack agents to twelve, increasing the scale of collaboration. Additionally, while the fault detection stage considers only healthy data and precision faults, the fault diagnosis stage introduces multiple fault classes, including drift and bias fault classes. These modifications enable a more comprehensive evaluation of how the collaboration improves fault diagnosis accuracy.

It is also important to note that, in this experiment, it is assumed that faults have already been detected correctly by the agents. Thus, the focus shifts to examining how collaboration among rack agents enhances diagnosis accuracy. By leveraging the CF* approach, an agent with limited diagnosis confidence can request support from compatible agents and interact and exchange data to refine its diagnosis based on the additional insights obtained. These adaptations ensure that the case study remains consistent while enabling investigation into the benefits of collaboration in fault diagnosis.

5.4.1 Dataset Elaboration for Fault Diagnosis Experiment

The temperature and humidity data from the ARENA racks were obtained using fourteen DHT22 sensors in a controlled room with temperatures ranging from 17 to 20 $^{\circ}C$ and humidity levels between 35% and 40%. Each temperature and humidity observation was recorded within a 10-second interval. Data from twelve sensors were used to train the diagnosis models of the twelve rack agents, each trained exclusively on data from its respective rack, resulting in distinct diagnosis models. The remaining two sensors were used to create two data samples to test the diagnosis performance of each agent.

Since all the sensors used in the experiments were healthy and the collected data had no faults, the data processing was conducted on the ESP32 associated with each rack to inject three categories of faults into the sensor outputs. The fault categories are precision, drift, and bias, each exhibiting a trend that distinguishes it from expected sensor behavior. The work developed by Jan *et al.* (2021) inspired the fault injection models.

Under normal conditions, i.e., without faults, the DHT22 sensors provide temperature and humidity data according to the previously introduced mathematical model presented in Equation (16), where $h(t)$ and $s(t)$ are the sensor output (temperature or humidity) without noise or fault and sensor data output respectively, expected in time t of healthy sensors, and $\eta \sim N(0,\delta_{\eta}^{2})$ is the noise with zero mean and low variance associated with each sensor.

The precision fault is related to sensor precision. This fault occurs when precision degradation is injected according to the mathematical model previously presented in Equation (17). The disturbance related to this fault has zero mean and high variance. The precision fault occurs due to physical connection problems in the circuit, noise resulting from high-frequency procedures, and physical damage to the sensor (JAN *et al.*, 2021).

The drift fault has a characteristic behavior where its output increases continuously at a constant rate, which is associated with the corrosion of the sensor material over time. The Equation (18) presents the mathematical model for this fault:

$$s_{drift}(t) = h(t) + \eta + b(t), \quad b(t) = b(t-1) + constant \tag{18}$$

where $b(t)$ is the drift added to the sensor output at time t, increasing over time at a constant rate, resulting in an increment in the sensor output.

Faults with characteristics of bias behavior have a constant value added to the sensor

output, and, as a result, a shift from the expected sensor output is observed. The Equation (19) expresses this characteristic of the bias fault. This behavior is associated with sensors with hardware problems, such as incorrect calibration or damaged components due to short circuits (JAN *et al.*, 2021).

$$s_{bias}(t) = h(t) + \eta + constant \tag{19}$$

Based on these fault models, the dataset for the fault diagnosis experiments was composed of the three fault categories. Table 19 presents the dataset distribution with the data divided between data training[3] and data test. In total, there are twelve training samples with drift, bias, and precision faults, and each sample is used to train the diagnosis models of a specific agent. The two remaining samples were used for experimental tests to evaluate and compare the performance of the agent models with local or collaborative approaches for fault diagnosis.

Table 19 – Training and test dataset for diagnosis of DHT22 sensor.

Dataset	Samples	Fault category	Amount of observations
data training	12 (one for each agent)	Drift	12000 (1000 for each sample)
		Bias	12000 (1000 for each sample)
		Precision	12000 (1000 for each sample)
data test	2 (one for each test)	Drift	1044 (522 for each sample)
		Bias	1162 (581 for each sample)
		Precision	1162 (581 for each sample)

Figure 68 presents the data test used for experiments with the same faults but manifested with different intensities. Since the diagnosis is only performed after the fault detection, the data used in the experiments corresponds only to the gray area of the dataset (in accordance with Table 19), i.e., the healthy data was not used for training and test the model.

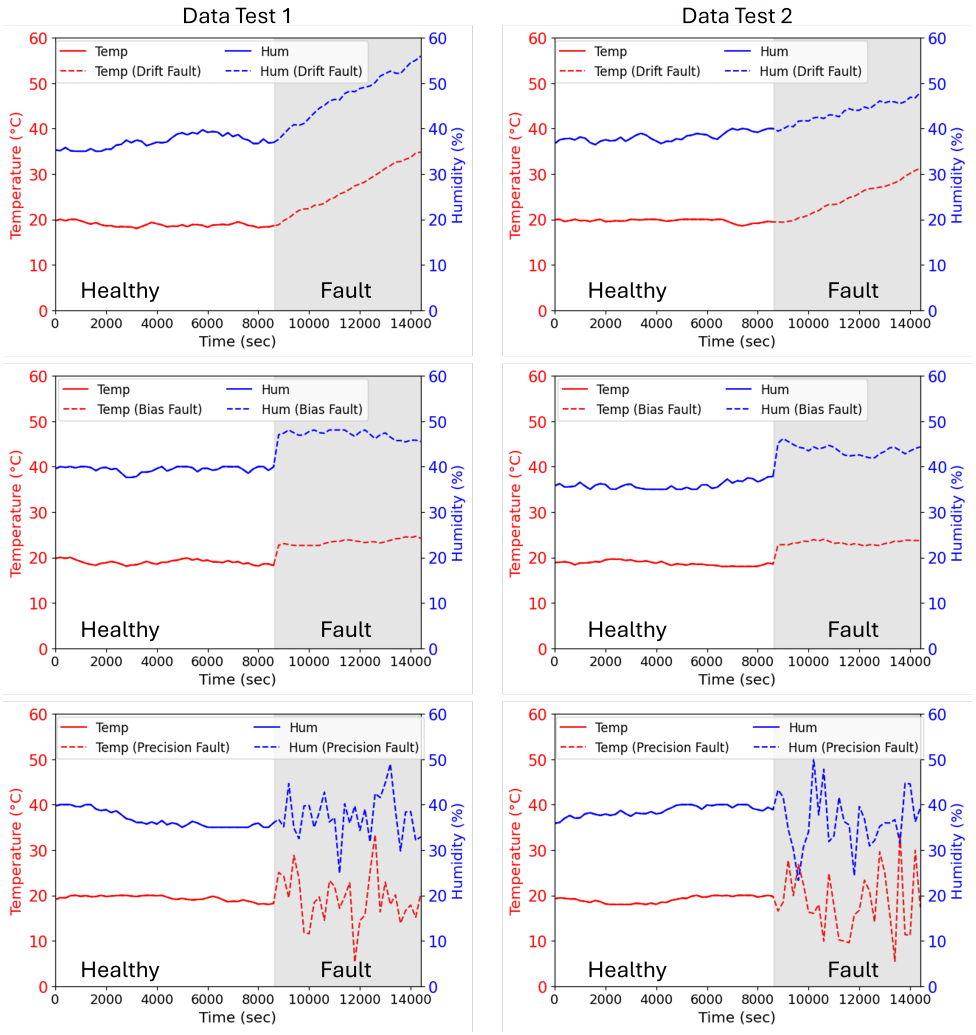
5.4.2 Experimental Procedure

The experiments were designed to evaluate the impact of collaboration on fault diagnosis, specifically whether it improves the ability to classify a fault category correctly. Therefore, each agent's individual fault diagnosis stage (i.e., LFT approach) was compared with the collaborative fault diagnosis based on the CF* specifications adopting the fault diagnosis protocol (see Figure 31 to check the protocol), as depicted in the flowchart of Figure 69.

The data-driven method based on ML techniques was adopted for the diagnosis stage due to its strong generalization capability for diagnosing faults, particularly faults in temperature

[3] Visualization of the training dataset in Appendix E

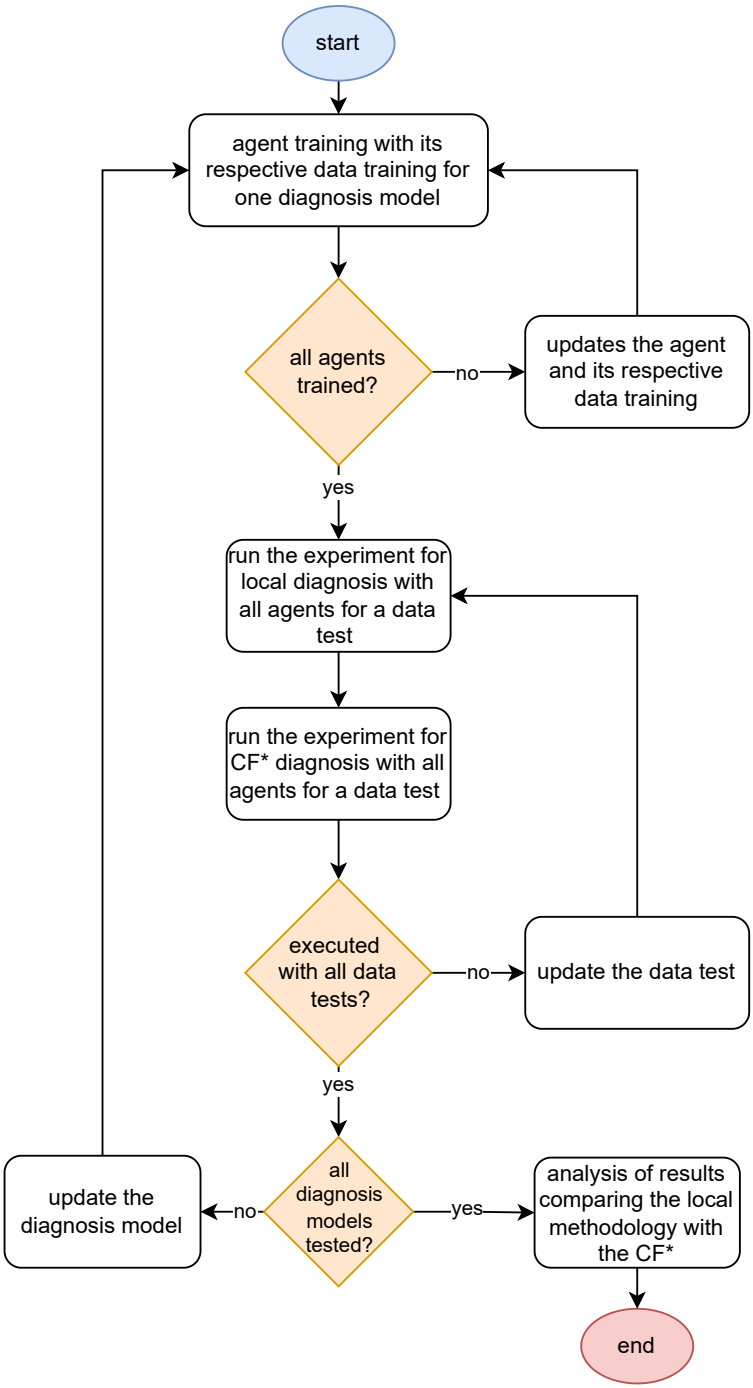
Figure 68 – Data test for diagnosis experiments with faults highlighted in the gray area.



and humidity sensors. Five ML algorithms were used in the experiments to assess the impact of collaboration on different algorithms (evaluated with two distinct data tests), namely: SVM, Nu-Support Vector Machine (NuSVM), Multi-Layer Perceptron (MLP), Random Forest (RF), and Gaussian Naive Bayes (GNB). These algorithms were chosen because they are widely used and representative in the context of fault diagnosis, as demonstrated in studies such as (ZIDI *et al.*, 2017; JAN *et al.*, 2017).

The diagnosis models were implemented in each agent using the Python library scikit-learn (PEDREGOSA *et al.*, 2011). For all ML models, the inputs consist of a sliding window containing sixty temperature and humidity observations, and the output categories were drift, bias, or precision fault classes. The ML algorithms (SVM, NuSVM, MLP, RF, and GNB) were trained with their respective default parameters according to the scikit-learn library. To obtain the similarity, i.e., the confidence of each classification, the inherent probability of each model was used, calibrated with the Plat Scaling method, available in the scikit-learn library, to obtain a

Figure 69 – Proposed procedure to compare the LFT and CF* for the fault diagnosis stage.



more accurate probability (NICULESCU-MIZIL; CARUANA, 2005).

To deploy the CF* approach, each agent performed self-fault diagnosis and collaboration as needed, based on the collaboration parameters τ, σ, and ρ, as presented in Table 20. These parameters were applied identically across all agents. The collaboration was triggered when the similarity measure was below the threshold $\tau = 0.90$. Due to their compatibility, the agent selection parameter σ included all twelve agent racks, allowing each agent to request and collaborate with all other agents in the system. The decision-making parameter ρ was based on a majority vote, meaning the fault category most frequently diagnosed by the agents was selected.

Table 20 – Instantiation strategies for the collaboration parameters of CF* in fault diagnosis experiments.

CF* Parameter	Instantiation Strategy Adopted	Value
Trigger (τ)	uncertainty or inconclusiveness for all models inference	confidence <90%
Selection (σ)	compatibility	collaborate with all rack agents available.
Decision-making (ρ)	most voted	fault detected or fault-free based on majority of votes

5.4.3 Evaluation Metrics

The performance metric adopted to evaluate fault diagnosis based on ML models is the F1-score (on a scale from 0% to 100%). This metric is the harmonic mean between precision and recall, capturing their trade-off. Therefore, a higher F1-score indicates fewer misclassifications by the fault diagnosis models.

The Wilcoxon signed-ranks test for paired samples (HOWELL, 1992) was selected to evaluate the influence of collaboration in the fault diagnosis performance of twelve agents. The Wilcoxon test was selected because the experimental results did not meet the requirements for the usual T-test for two paired samples, such as normal distribution of data (checked with the Shapiro-Wilk test) and homogeneous variance (checked with the Levene test). The Wilcoxon signed-ranks test for paired samples compares whether the median difference between the samples is the same (i.e., null hypothesis) or whether the difference between the medians is different (alternative hypothesis). For more details of this statistical test, it is recommended to see Chapters 18.6 and 18.7 of (HOWELL, 1992).

5.4.4 Results for Diagnosis Stage

Table 21 details the F1-Score results for the data test 1 for the diagnosis experiment (see the left graphs of Figure 68) for including twelve agents using the different diagnosis models, considering the local and collaborative methodology based on CF*. These F1-score values represent the ability to classify bias, drift, and precision faults to which the temperature and humidity sensors are susceptible. By analyzing the mean and median values in Table 21, it is evident that the collaborative approach proposed in CF* improves the fault diagnosis performance across all diagnosis models compared to the local approach. A substantial example is the mean value of the agents trained with the MLP, which increased from 79% to 90%, demonstrating a significant improvement in the fault diagnosis performance.

Table 21 – Result for diagnosis in terms of F1-Score (%) for data test 1.

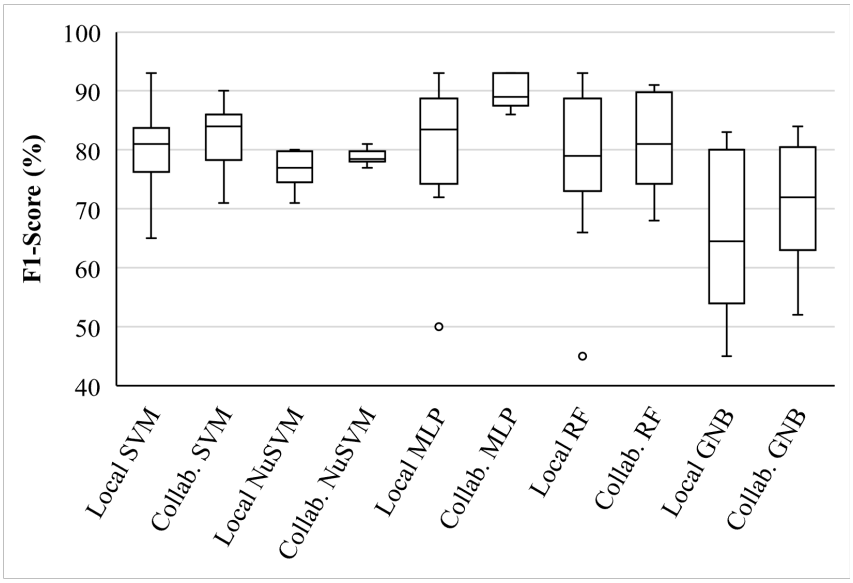
	Local SVM	**Collab. SVM**	**Local NuSVM**	**Collab. NuSVM**	**Local MLP**	**Collab. MLP**	**Local RF**	**Collab. RF**	**Local GNB**	**Collab. GNB**
$agent_1$	81	86	79	79	89	93	81	82	46	52
$agent_2$	89	87	80	79	84	87	73	75	68	71
$agent_3$	76	77	76	78	72	93	77	79	54	56
$agent_4$	81	84	71	77	93	93	73	80	58	73
$agent_5$	81	84	77	79	81	86	90	91	74	69
$agent_6$	77	77	71	78	90	89	83	84	83	83
$agent_7$	81	83	74	78	50	93	93	91	80	81
$agent_8$	83	86	76	78	83	89	91	91	80	79
$agent_9$	84	86	79	80	83	89	66	68	61	63
$agent_{10}$	93	90	80	80	88	89	85	86	83	84
$agent_{11}$	65	71	77	81	86	93	73	74	54	63
$agent_{12}$	71	82	80	78	50	86	45	72	45	75
mean	80.17	82.75	76.67	78.75	79.08	90.00	77.50	81.08	65.50	70.75
median	81	84	77	79	84	89	79	81	65	72

When evaluating the impact of collaboration on an agent from an individual perspective, it can be seen, for example, that the $agent_7$ trained with the MLP model significantly improved its performance. Without collaboration, its F1-score was 50%, but with collaboration, it reached 93%, matching the best result from all agents. However, collaboration sometimes introduces errors, negatively affecting some agents' performance. For instance, the MLP $agent_6$ slightly decreased its F1-score from 90% to 89%, and the SVM $agent_2$ dropped from 89% to 87%. These small reductions may be due to suboptimal settings for the collaboration parameters (τ, σ, ρ), indicating a need for a dedicated adjustment to ensure collaboration benefits all agents, for example, adopting the 7CM.

Figure 70 compares the fault diagnosis performance between the LFT and CF* approach for the diagnosis data test 1 using a box plot. Each box plot represents the performance of the

twelve agents under both methodologies with different diagnosis models. The plots show that collaboration improves the diagnosis capacity, as evidenced by the increased minimum values for all models and a decreased interquartile range, particularly in the NuSVM, MLP, and GNB models.

Figure 70 – Box plot comparing the diagnosis experiments based on LFT with CF* with data test 1.



To eliminate uncertainty in the achieved results, the experiments were repeated, using the data test 2 (see this data on graphs of Figure 68). Table 22 presents the F1-Score results for each agent submitted to this test in detail. Consistent with the previous findings, both the mean and median F1-Scores were higher when agents employed collaborative behavior for the fault diagnosis.

Table 22 – Result for diagnosis in terms of F1-Score (%) for data test 2.

	Local SVM	**Collab. SVM**	**Local NuSVM**	**Collab. NuSVM**	**Local MLP**	**Collab. MLP**	**Local RF**	**Collab. RF**	**Local GNB**	**Collab. GNB**
$agent_1$	82	82	78	76	64	80	72	74	80	81
$agent_2$	82	88	74	78	56	61	67	69	73	75
$agent_3$	78	79	75	76	77	78	69	70	80	80
$agent_4$	84	86	74	77	79	80	75	78	59	80
$agent_5$	82	83	76	77	78	77	79	78	78	80
$agent_6$	71	71	62	77	74	76	78	77	73	80
$agent_7$	81	83	77	78	55	77	74	82	80	80
$agent_8$	79	90	76	77	77	77	92	92	73	80
$agent_9$	85	86	77	75	54	67	68	68	67	67
$agent_{10}$	68	68	75	81	61	65	72	71	68	80
$agent_{11}$	75	78	74	77	78	77	71	73	75	75
$agent_{12}$	75	83	77	77	70	73	62	66	77	78
mean	78.50	81.42	74.58	77.17	68.58	74.00	73.25	74.83	73.58	78.00
median	80	83	76	77	72	77	72	74	74	80

Similarly, Figure 71 compares the fault diagnosis performance between the local and

collaborative methodology for data test 2 in the form of a box plot, indicating a trend of improvement in the fault diagnosis when agents collaborate.

Figure 71 – Box plot comparing the diagnosis experiments based on LFT with CF* with data test 2.

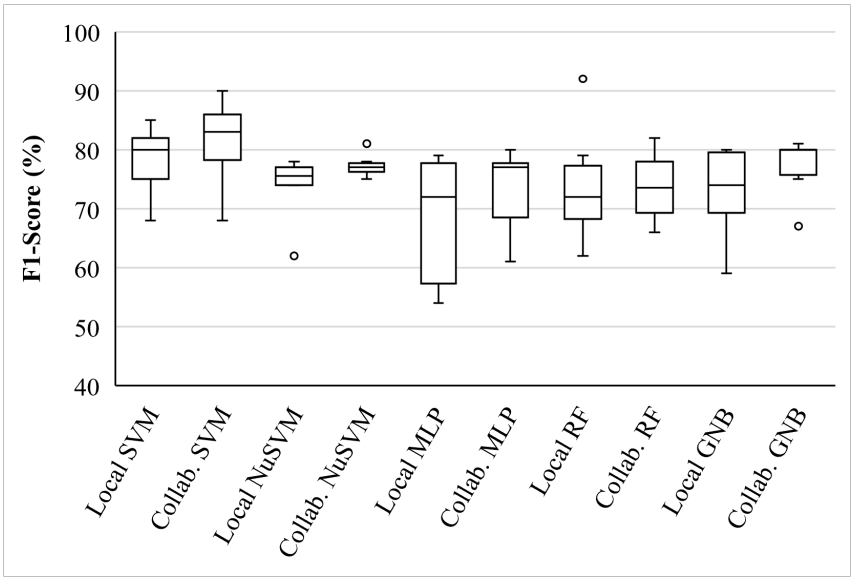
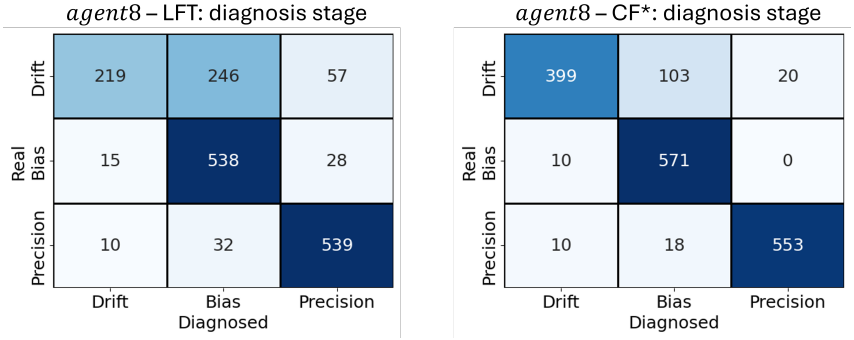


Figure 72 shows an example of the confusion matrix of an agent, in this case, selected the $agent_8$ trained with SVM model with data test 2. According to Table 22, $agent_8$ had a F1-score of 79% when performing the fault diagnosis locally. However, when collaborating, the F1-Score increases to 90%, which can be confirmed with the confusion matrix in Figure 72. Specifically, the collaboration notably enhanced the performance in diagnosing drift faults, increasing correct classifications from 219 to 399 classifications. For bias faults, correct classifications rose from 538 to 571 classifications, and for precision faults, there was also an improvement, with correct diagnoses increasing from 539 to 553 classifications.

Figure 72 – Comparison of confusion matrix of agent $agent_8$ using SVM model with diagnosis data test 2



5.4.5 Wilcoxon Test

To statistically validate the impact of collaboration for the presented experiments, the Wilcoxon signed-ranks test for paired samples was performed to verify whether there is a significant difference between the results obtained by the LFT and CF* approaches.

Thus, according to the Wilcoxon test, the following null hypothesis H_0 and alternative hypothesis H_1 are established respectively:

- H_0: The distribution of the medians of the F1-Score differences for fault diagnosis in the population (i.e., local vs. collaborative agent) is symmetric close to zero, i.e., indicating that any difference is due to chance.
- H_1: The distribution of the median F1-Score differences for local fault diagnosis in the population is smaller than the distribution of the median F1-Score differences for collaborative fault diagnosis in the population of agents.

The test results are presented in Table 23. The values of T^+ and T^- are statistics calculated for the Wilcoxon test from the difference between the pairs of samples (i.e., local and collaborative agent), where T^+ is the sum of the positive differences, and T^- is the sum of the negative differences. The statistic value W is the value used in the Wilcoxon test where the smaller between T^+ or T^- is selected. The p-value is obtained from the W statistic, and for $p - value < 0.05$ the null hypothesis H_0 is rejected with 95% of confidence.

Table 23 – Result of Wilcoxon Signed-Rank for Paired Samples test.

	Model	T^+	T^-	**W**	**p-value**	**Significance**
data test 1	SVM	9.5	56.5	9.5	0.0161	reject H_0
	NuSVM	9	46	9	0.0137	reject H_0
	MLP	1.5	64.5	1.5	0.0010	reject H_0
	RF	7.5	58.5	7.5	0.0093	reject H_0
	GNB	15	21	15	0.0181	reject H_0
data test 2	SVM	0	45	0	0,0020	reject H_0
	NuSVM	11	55	11	0.0269	reject H_0
	MLP	5	61	5	0.0049	reject H_0
	RF	7.5	47.5	7.5	0.0186	reject H_0
	GNB	0	36	0	0.0039	reject H_0

Therefore, according to Table 23, the fault diagnosis performance, in terms of F1-Score, for agents without collaboration (i.e. local) was lower than the performance of collaborative agents adopting the CF* approach. The Wilcoxon signed-rank test for paired samples showed

that this difference is statistically significant for the five diagnosis models tested considering the two different data tests.

5.5 CONJECTURE FOR RECOVERY STAGE FOR FAULTS IN THE RACK SENSOR

The previously presented experimental results demonstrate the effectiveness of the CF* approach in the fault detection and diagnosis stages of FT, highlighting how collaboration enhances the agents' ability to transcend their individual capabilities. The experiments focused on fault scenarios affecting temperature and humidity sensors in rack agents, which are responsible for managing and monitoring storage conditions. By leveraging collaborative fault detection, the proposed approach was compared against LFT and MCFT, demonstrating its capacity to improve fault detection accuracy. Similarly, in the fault diagnosis stage, CF* was evaluated against LFT, applying different diagnosis models, enabling agents to collaborate to classify faults precisely.

While the fault detection and diagnosis have been quantitatively analyzed, the fault recovery stage for temperature and humidity sensor faults is not experimentally assessed in this work. However, to conjecture on the potential implementation of fault recovery within the CF* approach for sensor faults in rack agents, consider the following scenario: once a rack agent detects and diagnoses a sensor fault, such as a precision fault leading to an inability to maintain the required temperature and humidity levels, the faulty rack agent notifies the stored products that its sensor is compromised and that it can no longer ensure proper storage conditions, aiming to mitigate the fault impact on these products, according to the fault impact mitigation protocol discussed in Figure 32.

Once the affected products are identified, the rack agent assumes the coordinator role in the collaborative recovery process, while the affected products act as followers (according to the collaborative recovery protocol illustrated in Figure 34). In response, these products initiate a recovery action, such as relocating to alternative storage and searching for available racks within the warehouse that can provide the required environmental conditions. This process involves storage negotiation with available rack agents and transport coordination with robot agents, as detailed in Appendix D, with further details in Figures 85 and 86. Then, the rack agent requests maintenance support to replace or fix the failed sensor.

This discussion provides insights into how collaborative fault recovery could be integrated into the CF* approach, extending the findings presented for the detection and diagnosis stages to the recovery stage for the fault in rack sensors.

5.6 SUMMARY

This Chapter discussed the case study, implementation, and experiments performed to evaluate the proposed approach in order to illustrate its applicability and contributions to deploying a collaborative FT in CPS. As discussed and illustrated, the workbench implements a CPS based on a distributed and decentralized MAS, leveraging the ROS infrastructure with agents programmed in Python.

The experiments were organized to focus primarily on the detection stage and later on the diagnosis stage. For each experiment, the datasets used were detailed, along with the experimental setup that instantiated agents with specific fault models.

The detection experiments focused on identifying precision faults in temperature and humidity sensors installed in rack agents. Five agent racks were deployed and trained with a fault-free and faulty sensor readings dataset. These experiments focused on comparing the precision, recall, and F1-score of traditional FT approaches, namely LFT and MCFT, with the proposed CF* approach. The results demonstrated the benefits of using collaboration, with the CF* approach significantly outperforming LFT and even surpassing MCFT, despite the latter being trained with a larger dataset.

Subsequently, an evaluation using the proposed 7CM was conducted to assess and adjust the collaboration performance. The experiment compared the behavior of five agents under three scenarios: without collaboration, with collaboration, and with collaboration adjusted according to the 7CM metrics analysis. The results clearly indicate that collaboration improves individual agent performance and that further performance gains can be achieved by tuning collaboration based on the insights provided by the metrics.

The experiments focused on the diagnosis stage involved twelve agent racks dedicated to diagnosing shift, bias, and precision faults in temperature and humidity sensors. The performance of the CF* approach was compared with the traditional LFT, with agents trained using five distinct ML models and tested against two data tests. The results showed consistent performance gains, with improvements of up to 10% in median performance (GNB model with data test 1). Moreover, the statistical validation using the Wilcoxon test confirmed that employing collaboration significantly outperformed the local strategy across all experiments.

Additionally, a conceptual discussion was presented regarding the recovery stage. Although not experimentally evaluated, this section explored how faults in sensor racks, such

as those identified in the detection and diagnosis stages, could be recovered collaboratively. The discussion considers how the agent with fault could orchestrate corrective actions among product agents to isolate the rack and reconfigure the storage without compromising the products accommodation. This stage will be further developed in future research.

This Chapter describes the experiments developed to answer the second research question of this thesis, the ***RQ-2***. The findings demonstrate that the collaboration between agents, each one potentially equipped with local intelligence and distinct models, for example, derived from different training data or algorithms, can outperform traditional approaches such as LFT and MCFT for the stages of detection and diagnosis. This diversity, when exploited through collaborative mechanisms, enhances FT for distributed and decentralized systems while preserving the system's scalability and flexibility required for CPS. Moreover, the ability to tune collaboration parameters with feedback of 7CM allows for adaptive behavior, further improving the collaboration behavior.

The proposed CF* approach can be implemented for broader applications in CPS. By instantiating agents with models tailored to fault scenarios in different contexts, the approach can be extended to various domains where collaboration among agents can enhance the robustness of FT while preserving flexibility, scalability, and robustness.

6 CONCLUSION AND FUTURE WORKS

This Chapter presents the main contributions from the research conducted in this thesis, highlighting how the proposed approach addresses the research questions and advances the state of the art in the domain of FT in CPS. A detailed list of publications derived from this research, which support and disseminate the contributions of this thesis, can be found in Appendix A.

The central research gap addressed in this work is the lack of a collaborative approach for FT in CPS modeled from a generic perspective, able to support diverse CPS applications and domains. Traditional FT approaches tend to be local and centralized, often rigid, and unable to exploit the full potential of agents deployed in dynamic, distributed CPS environments. In such contexts, agents must collaborate to overcome limitations from spatial or temporal myopia, unpredictable events, and interdependencies between CPS components.

To address this problem, the thesis proposes the CF*, a collaborative FT approach that integrates physical and cyber components under an agent meta-model. Rooted in MAS principles, CF* assigns each agent responsibility for a specific cyber-physical component and empowers them with autonomy and intelligence. These agents are capable of local decision-making for FT and, when necessary, initiate collaboration with compatible agents to enhance fault detection, diagnosis, and recovery stages.

The CF* formally defines when, who, and how an agent should collaborate, offering parameterized mechanisms for collaboration. Specific protocols were developed to promote interaction and collaboration during each FT stage, namely detection, diagnosis, and recovery, as well as for fault mitigation and recovery action involving a set of agents. These build upon standard agent communication languages such as FIPA-ACL, extending their application to fault management in CPS.

The dynamic behavior of agents operating under the CF* approach was formally described using the PN formalism. The fundamental PN was developed to model core agent functions, including control loop, interaction, collaborative FT, and model improvement. This PN was then refined to detail the collaborative behavior employed during fault detection, diagnosis, and recovery.

A case study was conducted using a mock-up warehouse (ARENA) to validate the proposed approach, focusing on fault detection and diagnosis in temperature and humidity sensors installed on storage racks. In the detection stage, performance metrics such as precision,

recall, and F1-score were compared across three approaches, namely LFT, and MCFT, and the proposed CF*. The results showed that CF* significantly outperformed LFT and even surpassed MCFT in many scenarios. During the diagnostic stage experiments, three distinct categories of faults, specifically shift, bias, and precision, were injected into the rack temperature and humidity sensor data. Agents trained with five distinct ML models, specifically SVM, NuSVM, MLP, RF, and GNB were evaluated using two testing datasets. The CF* consistently outperformed the LFT approach, yielding a median performance improvement of up to 10%. Statistical validation using the Wilcoxon test confirmed that the improvements observed with CF* were significant in all experiments.

However, the research also identifies that collaboration, if poorly managed, may lead to undesirable outcomes, such as redundant interaction or propagation of errors. To mitigate this, the 7CM metric was introduced to assess collaborative behaviors across multiple dimensions, including anxiety, collective error, insecurity, error induction, and correction. These metrics enable agents to fine-tune their collaborative FT strategies or retrain local FT models to achieve more effective and reliable outcomes.

The rest of this Chapter is organized as follows:

- Section 6.1 discusses the contributions derived from the work developed to answer the research questions.
- Section 6.2 presents a critical discussion of the limitations and boundaries of the proposed approach.
- Section 6.3 presented some perspectives for future work, briefly discussing some promising research opportunities that allow improving and enriching the proposed approach, with the aim of enhancing the robustness of FT in the context of CPS.

6.1 MAIN CONTRIBUTIONS

The research questions defined in Section 1.3 served as the foundation for the work developed in this thesis. They guided the CF* approach presented in Chapter 3 and the formal modeling described in Chapter 4, both were supported by the literature review in Chapter 2. The research questions were also addressed through the case study and experiments to proof of concept detailed in Chapter 5. Accordingly, the main contributions that were achieved throughout

this work developed to achieve the objectives of this thesis and answer the research questions are discussed in the following:

RQ-1: *What requirements and features must be considered to conceptualize an approach that serves as a generic model for implementing FT in CPS addressing robustness and scalability requirements without compromising system dependability?*

The first research question identifies the fundamental requirements and features necessary to conceptualize a generic approach for implementing fault tolerance in CPS. The main contribution in this context was the definition of the CF* approach, which integrates interaction and collaboration mechanisms among cyber-physical components. The CF* approach was designed to enhance system robustness and dependability without compromising the flexibility and scalability of CPS.

To address this research question, the thesis provides the following contributions:

- The state of the art review (Chapter 2) enabled the identification of enabling technologies, namely MAS, IoT, and AI, that support the conceptualization of a collaborative FT approach. It also revealed critical research gaps in current FT strategies for CPS.
- Collaborative parameters for FT that define when, who, and how an agent should collaborate, allowing the agent to transcend its individual FT capabilities.
- An agent meta-model for CF* that formally organizes an agent's physical and cyber modules, including its FT capabilities in terms of detection, diagnosis, recovery, system model, and collaboration interfaces. This generic meta-model proposed can be instantiated for different CPS domains and applications, supporting different FT methods.
- The design of collaborative protocols for each stage of the FT, allowing agents to share observations and decisions, request collaboration, mitigate fault impact, and coordinate recovery actions when facing systemic or global fault.

RQ-2: *Under which circumstances does collaboration for FT between distinct components outperform traditional approaches?*

The second research question focused on understanding under which circumstances collaboration in fault tolerance outperforms traditional approaches. This question guided the design and validation of the collaborative approach implemented within CF*, and it was addressed through experimental analysis and formal modeling.

The key contributions in this context were:

- The formulation of the 7CM, a set of metrics to evaluate collaboration dynamics and outcomes (e.g., anxiety, uncertainty, error induction/correction), which supports the analysis and fine-tuning of collaboration strategies between agents.
- A Petri Net-based formal model that captures agents local and collaborative behavior, allowing agent properties to be validated and refined for collaboration logic and synchronization with the control loop of the cyber-physical component.
- Comprehensive experimental validation using the ARENA, where CF* was compared with traditional LFT and MCFT approaches. The results showed that collaboration improved detection and diagnosis robustness, improving the precision, recall, and F1-score metrics.
- The fault diagnosis experiments provided statistical evidence, based on the Wilcoxon signed-rank test, that the CF* approach significantly outperforms LFT strategies across all diagnosis models, reinforcing the hypothesis that collaborative FT can outperform local strategies in CPS environments.

An important insight from this research is that collaboration between agents for FT is particularly effective when they are considered singular entities, for example, trained with distinct FT models and different training datasets. Despite their singularity, agents can still support each other through interaction sharing and collaboration. This leads to the emergence of collaborative behavior for FT, where the strengths and experiences of different agents contribute to solving local problems and enhancing the overall system's resilience through the collaborative mechanisms proposed by CF*.

6.2 LIMITATIONS AND DISCUSSION

Despite the conceptual consistency and the promising results demonstrated by the CF* approach for FT in CPS, some limitations must be acknowledged to provide a critical and balanced view of its applicability and to define the boundaries of its use.

First, while the CF* model enables agents to be trained individually with customized datasets and decision models, its implementation assumes a shared communication infrastructure with compatible data semantics. In heterogeneous CPS environments, where agents may rely on

different platforms, protocols, or data representations, this assumption may limit interoperability. This reveals a research opportunity to explore strategies that ensure semantic alignment and middleware-level compatibility, enabling collaboration among agents with diverse technological bases.

Second, the collaborative nature of CF* introduces a trade-off when compared to local and centralized FT approaches. While local models are fast and resource-efficient, they are limited by the agent's perceptual scope. Centralized models benefit from a global view but often lack scalability and are vulnerable to single points of failure. In contrast, CF* leverages agent collaboration; however, this comes at the cost of increased communication overhead, potential latency in decision-making, and higher bandwidth usage, particularly in dense resource or constrained networks. The effectiveness of collaboration thus depends heavily on network conditions, the frequency of agent interactions, and the efficiency of the collaboration protocols.

Another important consideration is the trustworthiness and reliability of external agents involved in the collaboration process. The current CF* does not explicitly model trust or reputation mechanisms, assuming that all agents are honest and cooperative. However, in practical applications, agents can behave unreliably, maliciously, or erroneously, leading to the propagation of incorrect fault-related information, reduced diagnostic accuracy, or even unsafe recovery actions. This reveals another research opportunity: investigating trust assessment frameworks or reputation-based collaboration filters to mitigate such vulnerabilities.

Finally, the current implementation of CF* does not explicitly address real-time constraints. Although the model supports adaptive and decentralized decision-making, its suitability for time-critical applications has not yet been validated. A more rigorous analysis that involves time-sensitive modeling and formal verification would be necessary to guarantee its applicability in real-time or safety-critical systems.

These limitations do not undermine the relevance or originality of the CF* approach. On the contrary, they help define the boundaries of its applicability and highlight important technical challenges to be addressed in future research.

6.3 FUTURE WORK

The work presented in this thesis opens new research avenues in the field of collaborative FT for CPS and provides a solid foundation for future developments. While the proposed approach already demonstrates promising results, there are several opportunities to extend and

improve the CF* approach. This section highlights some of the most relevant and promising research directions, some of which are already under development for scientific dissemination, while others are pending the availability of resources. The future research is grouped into topics:

- **Explore others aspects of CF*, namely the recovery stage and FT model updating via the system module** *(ongoing)*:

  The recovery stage and model updating of CF* was defined and formulated in this thesis but not yet experimentally validated. Future work includes implementing and evaluating collaborative recovery strategies, exploring the agents interactions and proposed protocols to elaborate collaborative recovery action, and collaborative recovery execution. Also, explore the learning capability with the system model of CF* to update the models using the historic fault detection, diagnosis, and recovery based on collaboration feedback.

- **Instantiating CF* in different CPS domains and applications** *(ongoing)*:

  The CF* was validated in a specific case study to proof of concept. However, considering the generic and modular design that allows CF* for adaptation to a wide range of CPS domains. A key future direction involves instantiating CF* in different application scenarios, such as industrial automation, autonomous vehicles, and smart grids. This will help evaluate its versatility, scalability, and domain-specific performance. Exploring new contexts will also allow for identifying domain-dependent constraints and opportunities, further refining the meta-model and expanding its applicability across different CPS environments.

- **Modeling and validating real-time criteria of CF* for safety-critical CPS** *(future work)*:

  A relevant research direction involves modeling and validating the real-time constraints associated with the CF* approach, particularly for deployment in safety-critical CPS domains. The goal is to define a generic and adaptable model capable of supporting collaborative FT within strict temporal bounds required by real-time CPS applications. This generalized model aims to accommodate a range of systems with time-sensitive requirements, ensuring that collaboration in detection, diagnosis, and recovery processes can be executed predictably and reliably. Future work will explore formal methods and real-time validation techniques to guarantee that the CF* adheres to the timing constraints necessary for dependable operation in real-time environments.

- **Explore interoperability models for agents to investigate collaboration for FT in heterogeneous cyber-physical components** *(future work)*:

  One promising research direction lies in investigating interoperability mechanisms that enable heterogeneous agents representing different classes of cyber-physical components to collaborate effectively in the context of FT. Given that real-world CPSs often comprise components with diverse capabilities and communication protocols, this work aims to define interoperability models and integration strategies that support the exchange of information across heterogeneous agents. Such models are expected to enhance the collaborative capabilities of the CF* approach. This includes exploring middleware solutions, common ontologies, and protocol translation techniques to facilitate seamless collaboration to FT in heterogeneous CPS environments.

- **Integrating Digital Twin for diagnosis and recovery stages through simulation-based model refinement** *(future work)*:

  A promising research direction involves the integration of the digital twin paradigm into the diagnosis and recovery stages of the CF* approach. This integration aims to establish a continuous synchronization between the physical and virtual components of the system, enabling agents to learn from digitized data and update their fault models dynamically. By leveraging what-if simulations and optimization techniques within the digital twin environment, agents could proactively test alternative responses to potential faults and refine their diagnosis and recovery strategies. This would allow for more context-aware, adaptive, and resource-efficient decisions, ultimately enhancing the resilience and reliability of the system over time. Using simulation-based learning with operational feedback would also improve long-term FT through continuous model refinement.

- **Developing the Extended CF* incorporating fault prognostics capabilities** *(future work)*:

  Building upon the foundation established by the current CF* approach, a relevant research opportunity involves developing an Extended CF* version that integrates fault prognostics capabilities. While the current focus is on detection, diagnosis, and recovery, incorporating prognostics would allow agents to anticipate potential fault based on operational patterns, environmental factors, or degradation indicators, adopting collaboration to improve its prognostics. This predictive layer could enhance the robustness and reliability of CPS.

REFERENCES

ABAR, Sameera; THEODOROPOULOS, Georgios K; LEMARINIER, Pierre; O'HARE, Gregory MP. Agent based modelling and simulation tools: A review of the state-of-art software. **Computer Science Review**, Elsevier, v. 24, p. 13–33, 2017.

ADOLPHS, Peter; BEDENBENDER, Heinz; DIRZUS, Dagmar; EHLICH, Martin; EPPLE, Ulrich; HANKEL, Martin; HEIDEL, Roland; HOFFMEISTER, Michael; HUHLE, Haimo; KäRCHER, Bernd; KOZIOLEK, Heiko; PICHLER, Reinhold; POLLMEIER, Stefan; SCHEWE, Frank; WALTER, Armin; WASER, Bernd; WOLLSCHLAEGER, Martin. **Status Report - Reference Architecture Model Industrie 4.0 (RAMI4.0)**. 2015.

ALI, Mudassar; RIAZ, Nida; ASHRAF, Muhammad Ikram; QAISAR, Saad; NAEEM, Muhammad. Joint cloudlet selection and latency minimization in fog networks. **IEEE Transactions on Industrial Informatics**, IEEE, v. 14, n. 9, p. 4055–4063, 2018.

ALSHDADI, Abdulrahman A. Cyber-physical system with iot-based smart vehicles. **Soft Computing**, Springer, v. 25, n. 18, p. 12261–12273, 2021.

AMINIZADEH, Sarina; HEIDARI, Arash; TOUMAJ, Shiva; DARBANDI, Mehdi; NAVIMIPOUR, Nima Jafari; REZAEI, Mahsa; TALEBI, Samira; AZAD, Poupak; UNAL, Mehmet. The applications of machine learning techniques in medical data processing based on distributed computing and the internet of things. **Computer methods and programs in biomedicine**, Elsevier, v. 241, p. 107745, 2023.

ATTALLAH, Salma MA; FAYEK, Magda B; NASSAR, Salwa M; HEMAYED, Elsayed E. Proactive load balancing fault tolerance algorithm in cloud computing. **Concurrency and Computation: Practice and Experience**, Wiley Online Library, v. 33, n. 10, p. e6172, 2021.

AVIZIENIS, Algirdas; LAPRIE, J-C; RANDELL, Brian; LANDWEHR, Carl. Basic concepts and taxonomy of dependable and secure computing. **IEEE transactions on dependable and secure computing**, IEEE, v. 1, n. 1, p. 11–33, 2004.

BADR, Mustafa; TALEBI, Heidar Ali; KHOSRAVI, Mohammad A. A novel approach for discriminating faults and replay attacks in hybrid systems. **IEEE Access**, IEEE, 2024.

BARBOSA, José; LEITÃO, Paulo; ADAM, Emmanuel; TRENTESAUX, Damien. Dynamic self-organization in holonic multi-agent manufacturing systems: The adacor evolution. **Computers in industry**, Elsevier, v. 66, p. 99–111, 2015.

BARBOSA, José; LEITÃO, Paulo; TRENTESAUX, Damien; COLOMBO, Armando W; KARNOUSKOS, Stamatis. Cross benefits from cyber-physical systems and intelligent products for future smart industries. *In*: IEEE. **14th International Conference on Industrial Informatics (INDIN)**. 2016. p. 504–509.

BAYAR, Nawel; DARMOUL, Saber; HAJRI-GABOUJ, Sonia; PIERREVAL, Henri. Fault detection, diagnosis and recovery using Artificial Immune Systems: A review. **Engineering Applications of Artificial Intelligence**, Elsevier, v. 46, p. 43–57, 2015.

BEG, Omar Ali; NGUYEN, Luan Viet; JOHNSON, Taylor T; DAVOUDI, Ali. Cyber-physical anomaly detection in microgrids using time-frequency logic formalism. **IEEE Access**, IEEE, v. 9, p. 20012–20021, 2021.

BENMALEK, Mourad. Ransomware on cyber-physical systems: Taxonomies, case studies, security gaps, and open challenges. **Internet of Things and Cyber-Physical Systems**, Elsevier, v. 4, p. 186–202, 2024.

BERG, Jeroen P Van den; ZIJM, Willem HM. Models for warehouse management: Classification and examples. **International journal of production economics**, Elsevier, v. 59, n. 1-3, p. 519–528, 1999.

BORAL, Soumava; CHATURVEDI, Sanjay Kumar; NAIKAN, Vallayil Narayana Achutha. A case-based reasoning system for fault detection and isolation: a case study on complex gearboxes. **Journal of Quality in Maintenance Engineering**, Emerald Publishing Limited, v. 25, n. 2, p. 213–235, 2019.

CAO, Kun; HU, Shiyan; SHI, Yang; COLOMBO, Armando Walter; KARNOUSKOS, Stamatis; LI, Xin. A survey on edge and edge-cloud computing assisted cyber-physical systems. **IEEE Transactions on Industrial Informatics**, IEEE, v. 17, n. 11, p. 7806–7819, 2021.

CARDOSO, Janette; VALETTE, Robert. **Redes de petri**. : Editora da UFSC Florianópolis, SC:, 1997.

CASSANO, Luca; MIELE, Antonio; MIONE, Francesco; TONELLOTTO, Nicola; VALLATI, Carlo. Design of fault-tolerant distributed cyber–physical systems for smart environments. **IEEE Embedded Systems Letters**, IEEE, v. 14, n. 2, p. 79–82, 2021.

CHAKRAVARTY, Arghya; MAHANTA, Chitralekha. Actuator fault-tolerant control (ftc) design with post-fault transient improvement for application to aircraft control. **International Journal of Robust and Nonlinear Control**, Wiley Online Library, v. 26, n. 10, p. 2049–2074, 2016.

CHEN, Baotong; WAN, Jiafu; CELESTI, Antonio; LI, Di; ABBAS, Haider; ZHANG, Qin. Edge computing in iot-based manufacturing. **IEEE Communications Magazine**, IEEE, v. 56, n. 9, p. 103–109, 2018.

CHEN, Baotong; WAN, Jiafu; SHU, Lei; LI, Peng; MUKHERJEE, Mithun; YIN, Boxing. Smart factory of industry 4.0: Key technologies, application case, and challenges. **IEEE Access**, IEEE, v. 6, p. 6505–6519, 2017.

CHEN, Hong. Applications of cyber-physical system: a literature review. **Journal of Industrial Integration and Management**, World Scientific, v. 2, n. 03, p. 1750012, 2017.

CHEN, Kai-Ying; CHEN, Long-Sheng; CHEN, Mu-Chen; LEE, Chia-Lung. Using SVM based method for equipment fault detection in a thermal power plant. **Computers in industry**, Elsevier, v. 62, n. 1, p. 42–50, 2011.

CHEN, Zhelun; O'NEILL, Zheng; WEN, Jin; PRADHAN, Ojas; YANG, Tao; LU, Xing; LIN, Guanjing; MIYATA, Shohei; LEE, Seungjae; SHEN, Chou *et al.* A review of data-driven fault detection and diagnostics for building hvac systems. **Applied Energy**, Elsevier, v. 339, p. 121030, 2023.

CHENG, Jiangfeng; YANG, Yi; ZOU, Xiaofu; ZUO, Ying. 5g in manufacturing: a literature review and future research. **The International Journal of Advanced Manufacturing Technology**, Springer, v. 131, n. 11, p. 5637–5659, 2024.

CHERAGHLOU, Mehdi Nazari; KHADEM-ZADEH, Ahmad; HAGHPARAST, Majid. A survey of fault tolerance architecture in cloud computing. **Journal of Network and Computer Applications**, Elsevier, v. 61, p. 81–92, 2016.

CHI, Yuanfang; DONG, Yanjie; WANG, Z Jane; YU, F Richard; LEUNG, Victor CM. Knowledge-based fault diagnosis in industrial internet of things: a survey. **IEEE Internet of Things Journal**, IEEE, v. 9, n. 15, p. 12886–12900, 2022.

CHIANG, Mung; ZHANG, Tao. Fog and iot: An overview of research opportunities. **IEEE Internet of things journal**, IEEE, v. 3, n. 6, p. 854–864, 2016.

COLABIANCHI, Silvia; COSTANTINO, Francesco; GRAVIO, Giulio Di; NONINO, Fabio; PATRIARCA, Riccardo. Discussing resilience in the context of cyber physical systems. **Computers & Industrial Engineering**, Elsevier, v. 160, p. 107534, 2021.

COLOMBO, AW; CARELLI, R. Petri nets for designing manufacturing systems. *In*: **Computer-Assisted Management and Control of Manufacturing Systems**. : Springer, 1997. p. 297–324.

COLOMBO, Armando Walter; CARELLI, Ricardo; KUCHEN, Benjamin. A temporised petri net approach for design, modelling and analysis of flexible production systems. **The International Journal of Advanced Manufacturing Technology**, Springer, v. 13, p. 214–226, 1997.

COLOMBO, Armando W; KARNOUSKOS, Stamatis; KAYNAK, Okyay; SHI, Yang; YIN, Shen. Industrial cyberphysical systems: A backbone of the fourth industrial revolution. **IEEE Industrial Electronics Magazine**, IEEE, v. 11, n. 1, p. 6–16, 2017.

DORRI, Ali; KANHERE, Salil S; JURDAK, Raja. Multi-agent systems: A survey. **IEEE Access**, IEEE, v. 6, p. 28573–28593, 2018.

DOWDESWELL, Barry; SINHA, Roopak; MACDONELL, Stephen G. Finding faults: A scoping study of fault diagnostics for Industrial Cyber-Physical Systems. **Journal of systems and software**, Elsevier, v. 168, p. 110638, 2020.

EGWUTUOHA, Ifeanyi P; LEVY, David; SELIC, Bran; CHEN, Shiping. A survey of fault tolerance mechanisms and checkpoint/restart implementations for high performance computing systems. **The Journal of Supercomputing**, Springer, v. 65, p. 1302–1326, 2013.

FEI, Xiang; SHAH, Nazaraf; VERBA, Nandor; CHAO, Kuo-Ming; SANCHEZ-ANGUIX, Victor; LEWANDOWSKI, Jacek; JAMES, Anne; USMAN, Zahid. Cps data streams analytics based on machine learning for cloud and fog computing: A survey. **Future generation computer systems**, Elsevier, v. 90, p. 435–450, 2019.

FERBER, Jacques; WEISS, Gerhard. **Multi-agent systems: an introduction to distributed artificial intelligence**. : Addison-wesley Reading, 1999. v. 1.

FIDELIS, Samuel Amico; CASTRO, Márcio; SIQUEIRA, Frank. Distributed learning using consensus on edge ai. *In*: IEEE. **2022 XII Brazilian Symposium on Computing Systems Engineering (SBESC)**. 2022. p. 1–8.

GARCÍA-ESTEBAN, Juan Alberto; PIARDI, Luis; LEITÃO, Paulo; CURTO, Belén; MORENO, Vidal. An interaction strategy for safe human co-working with industrial collaborative robots. *In*: IEEE. **2021 4th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS)**. 2021. p. 585–590.

GARRAGHAN, Peter; PERKS, Stuart; OUYANG, Xue; MCKEE, David; MORENO, Ismael Solis. Tolerating transient late-timing faults in cloud-based real-time stream processing. *In*: IEEE. **2016 IEEE 19th International Symposium on Real-Time Distributed Computing (ISORC)**. 2016. p. 108–115.

GENSH, Rem; RAFIEV, Ashur; ROMANOVSKY, Alexander; GARCIA, Alessandro; XIA, Fei; YAKOVLEV, Alex. Architecting holistic fault tolerance. *In*: IEEE. **2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)**. 2017. p. 5–8.

GRETHLER, Michael; MARINOV, Marin B; KLUMPP, Vesa. Embedded machine learning for machine condition monitoring. *In*: SPRINGER. **International Conference on Future Access Enablers of Ubiquitous and Intelligent Infrastructures**. 2021. p. 217–228.

GRIFFOR, Edward R; GREER, Christopher; WOLLMAN, David A; BURNS, Martin J. Framework for cyber-physical systems: Volume 1, overview. Special Publication (NIST SP), National Institute of Standards and Technology, Gaithersburg, MD, 2017.

GUNES, Volkan; PETER, Steffen; GIVARGIS, Tony; VAHID, Frank. A survey on concepts, applications, and challenges in cyber-physical systems. **KSII Transactions on Internet and Information Systems (TIIS)**, Korean Society for Internet Information, v. 8, n. 12, p. 4242–4268, 2014.

GUPTA, Nirupam; VAIDYA, Nitin H. Fault-tolerance in distributed optimization: The case of redundancy. *In*: **Proceedings of the 39th Symposium on Principles of Distributed Computing**. 2020. p. 365–374.

GUPTA, Priyanka; RANVIJAY. Enhanced energy-aware fault tolerance technique for real-time task on heterogeneous multicore system. *In*: SPRINGER. **International Conference on VLSI, Communication and Signal processing**. 2022. p. 745–762.

HA, Nam; XU, Kai; REN, Guanghui; MITCHELL, Arnan; OU, Jian Zhen. Machine learning-enabled smart sensor systems. **Advanced Intelligent Systems**, Wiley Online Library, v. 2, n. 9, p. 2000063, 2020.

HABIB, Maki K; CHIMSOM, Chukwuemeka. Cps: Role, characteristics, architectures and future potentials. **Procedia Computer Science**, Elsevier, v. 200, p. 1347–1358, 2022.

HAN, Yunlong. Wavelet analysis and consensus algorithm-based fault-tolerant control for smart grids. **Frontiers in Energy Research**, Frontiers Media SA, v. 11, p. 1160256, 2023.

HASEEB, Khalid; SABA, Tanzila; REHMAN, Amjad; AHMED, Zara; SONG, Houbing Herbert; WANG, Huihui Helen. Trust management with fault-tolerant supervised routing for smart cities using internet of things. **IEEE Internet of Things Journal**, IEEE, v. 9, n. 22, p. 22608–22617, 2022.

HEHENBERGER, Peter; VOGEL-HEUSER, Birgit; BRADLEY, David; EYNARD, Benoît; TOMIYAMA, Tetsuo; ACHICHE, Sofiane. Design, modelling, simulation and integration of

cyber physical systems: Methods and applications. **Computers in Industry**, Elsevier, v. 82, p. 273–289, 2016.

HERNANDEZ, DM; PERALTA, G; MANERO, L; GOMEZ, R; BILBAO, J; ZUBIA, C. Energy and coverage study of lpwan schemes for industry 4.0. *In*: IEEE. **2017 IEEE International Workshop of Electronics, Control, Measurement, Signals and their Application to Mechatronics (ECMSM)**. 2017. p. 1–6.

HOUSH, Mashor; OHAR, Ziv. Model-based approach for cyber-physical attack detection in water distribution systems. **Water research**, Elsevier, v. 139, p. 132–143, 2018.

HOWELL, David C. **Statistical methods for psychology**. : PWS-Kent Publishing Co, 1992.

HU, Fei; LU, Yu; VASILAKOS, Athanasios V; HAO, Qi; MA, Rui; PATIL, Yogendra; ZHANG, Ting; LU, Jiang; LI, Xin; XIONG, Neal N. Robust cyber–physical systems: Concept, models, and implementation. **Future generation computer systems**, Elsevier, v. 56, p. 449–475, 2016.

HU, Pengfei; DHELIM, Sahraoui; NING, Huansheng; QIU, Tie. Survey on fog computing: architecture, key technologies, applications and open issues. **Journal of network and computer applications**, Elsevier, v. 98, p. 27–42, 2017.

HUANG, Yuanyuan; LU, Xin. **Security, governance, and challenges of the new generation of cyber-physical-social systems**. : Frontiers Media SA, 2024. 1464919 p.

ISERMANN, R. **Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance**. : Springer Science & Business Media, 2006.

JAN, Sana Ullah; LEE, Young Doo; KOO, In Soo. A distributed sensor-fault detection and diagnosis framework using machine learning. **Information Sciences**, Elsevier, v. 547, p. 777–796, 2021.

JAN, Sana Ullah; LEE, Young-Doo; SHIN, Jungpil; KOO, Insoo. Sensor fault classification based on support vector machine and statistical time-domain features. **IEEE Access**, IEEE, v. 5, p. 8682–8690, 2017.

JANUÁRIO, Fábio; CARDOSO, Alberto; GIL, Paulo. A distributed multi-agent framework for resilience enhancement in cyber-physical systems. **IEEE Access**, IEEE, v. 7, p. 31342–31357, 2019.

JHA, Amitkumar Vidyakant; APPASANI, Bhargav; GHAZALI, Abu Nasar; PATTANAYAK, Prabina; GURJAR, Devendra Singh; KABALCI, Ersan; MOHANTA, DK. Smart grid cyber-physical systems: Communication technologies, standards and challenges. **Wireless Networks**, Springer, v. 27, n. 4, p. 2595–2613, 2021.

JIANG, Yuchen; YIN, Shen; KAYNAK, Okyay. Data-driven monitoring and safety control of industrial cyber-physical systems: Basics and beyond. **IEEE Access**, IEEE, v. 6, p. 47374–47384, 2018.

JING, Guanlin; ZOU, Yifei; YU, Dongxiao; LUO, Chuanwen; CHENG, Xiuzhen. Efficient fault-tolerant consensus for collaborative services in edge computing. **IEEE Transactions on Computers**, IEEE, v. 72, n. 8, p. 2139–2150, 2023.

KAGERMANN, Henning; WAHLSTER, Wolfgang; HELBIG, Johannes. Securing the future of german manufacturing industry: Recommendations for implementing the strategic initiative industrie 4.0. **Final report of the Industrie**, v. 4, n. 0, 2013.

KALASKAR, Chetankumar; SOMASUNDARAM, Thangam. Enhancing fault tolerance: Dual q-learning with dynamic scheduling. **Indones. J. Electr. Eng. Comput. Sci**, v. 33, p. 1150, 2024.

KALEMPA, Vivian Cremer; PIARDI, Luis; LIMEIRA, Marcelo; OLIVEIRA, André Schneider De. Fault-resilient collective ternary-hierarchical behavior to smart factories. **IEEE Access**, IEEE, v. 8, p. 176905–176915, 2020.

KARNOUSKOS, Stamatis; COLOMBO, Armando Walter; BANGEMANN, Thomas; MANNINEN, Keijo; CAMP, Roberto; TILLY, Marcel; STLUKA, Petr; JAMMES, François; DELSING, Jerker; ELIASSON, Jens. A soa-based architecture for empowering future collaborative cloud-based industrial automation. *In*: IEEE. **38th Annual Conference on IEEE Industrial Electronics Society (IECON)**. 2012. p. 5766–5772.

KARNOUSKOS, Stamatis; LEITAO, Paulo. Key contributing factors to the acceptance of agents in industrial environments. **IEEE Transactions on Industrial Informatics**, IEEE, v. 13, n. 2, p. 696–703, 2016.

KARNOUSKOS, Stamatis; LEITAO, Paulo; RIBEIRO, Luis; COLOMBO, Armando Walter. Industrial agents as a key enabler for realizing industrial cyber-physical systems: Multiagent systems entering industry 4.0. **IEEE Industrial Electronics Magazine**, IEEE, v. 14, n. 3, p. 18–32, 2020.

KHAITAN, Siddhartha Kumar; MCCALLEY, James D. Design techniques and applications of cyberphysical systems: A survey. **IEEE systems journal**, IEEE, v. 9, n. 2, p. 350–365, 2014.

KHALID, Waqas; YU, Heejung; ALI, Rashid; ULLAH, Rehmat. Advanced physical-layer technologies for beyond 5g wireless communication networks. **Sensors**, MDPI, v. 21, n. 9, p. 3197, 2021.

KHAN, Maqbool; WU, Xiaotong; XU, Xiaolong; DOU, Wanchun. Big data challenges and opportunities in the hype of industry 4.0. *In*: IEEE. **2017 IEEE International Conference on Communications (ICC)**. 2017. p. 1–6.

KONG, Xiangyu; ZHANG, Tong. Distributed cooperative sliding mode fault-tolerant control for multiple high-speed trains based on actor-critic neural network. **Journal of Mathematics**, Wiley Online Library, v. 2021, n. 1, p. 9943170, 2021.

KOREN, Israel; KRISHNA, C Mani. **Fault-tolerant systems**. : Morgan Kaufmann, 2020.

KUMAR, Sunil; BHOWMIK, Biswajit. Emergence, evolution, and applications of cyber-physical systems in smart society. *In*: IEEE. **2024 Fourth International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)**. 2024. p. 1–8.

LAPRIE, Jean-Claude. Dependability: Basic concepts and terminology. *In*: **Dependability: Basic Concepts and Terminology**. : Springer, 1992. p. 3–245.

LAPRIE, Jean-Claude. Dependability—its attributes, impairments and means. *In*: **Predictably Dependable Computing Systems**. : Springer, 1995. p. 3–18.

LEE, Edward A. Cyber physical systems: Design challenges. *In*: IEEE. **11th international symposium on object and component-oriented real-time distributed computing (ISORC)**. 2008. p. 363–369.

LEE, Jay; BAGHERI, Behrad; KAO, Hung-An. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. **Manufacturing letters**, Elsevier, v. 3, p. 18–23, 2015.

LEE, Jay; DAVARI, Hossein; SINGH, Jaskaran; PANDHARE, Vibhor. Industrial artificial intelligence for industry 4.0-based manufacturing systems. **Manufacturing letters**, Elsevier, v. 18, p. 20–23, 2018.

LEITÃO, Paulo; COLOMBO, Armando Walter; KARNOUSKOS, Stamatis. Industrial automation based on cyber-physical systems technologies: Prototype implementations and challenges. **Computers in industry**, Elsevier, v. 81, p. 11–25, 2016.

LEITAO, Paulo; KARNOUSKOS, Stamatis; RIBEIRO, Luis; LEE, Jay; STRASSER, Thomas; COLOMBO, Armando W. Smart agents in industrial cyber–physical systems. **Proceedings of the IEEE**, IEEE, v. 104, n. 5, p. 1086–1101, 2016.

LEITÃO, Paulo; KARNOUSKOS, Stamatis; STRASSER, Thomas I; JIA, Xiaodong; LEE, Jay; COLOMBO, Armando Walter. Alignment of the ieee industrial agents recommended practice

standard with the reference architectures rami4. 0, iira, and sgam. **IEEE Open Journal of the Industrial Electronics Society**, IEEE, v. 4, p. 98–111, 2023.

LEITÃO, Paulo; MENDES, J Marco; BEPPERLING, Axel; CACHAPA, Daniel; COLOMBO, Armando W; RESTIVO, Francisco. Engineering tools for the integration of service-oriented production systems. **IFAC Proceedings Volumes**, Elsevier, v. 42, n. 4, p. 1772–1777, 2009.

LEITÃO, Paulo; RESTIVO, Francisco. Adacor: A holonic architecture for agile and adaptive manufacturing control. **Computers in industry**, Elsevier, v. 57, n. 2, p. 121–130, 2006.

LEITÃO, Paulo Jorge Pinto. **An Agile and Adaptive Holonic Architecture for Manufacturing Control**. 2004. Phd Thesis (PhD Thesis) — Faculdade de Engenharia da Universidade do Porto (FEUP), 2004.

LENG, Bohan; GAO, Shuo; XIA, Tangbin; PAN, Ershun; SEIDELMANN, Joachim; WANG, Hao; XI, Lifeng. Digital twin monitoring and simulation integrated platform for reconfigurable manufacturing systems. **Advanced Engineering Informatics**, Elsevier, v. 58, p. 102141, 2023.

LESCH, Veronika; ZÜFLE, Marwin; BAUER, André; IFFLÄNDER, Lukas; KRUPITZER, Christian; KOUNEV, Samuel. A literature review of iot and cps—what they are, and what they are not. **Journal of Systems and Software**, Elsevier, v. 200, p. 111631, 2023.

LEZOCHE, Mario; PANETTO, Hervé. Cyber-physical systems, a new formal paradigm to model redundancy and resiliency. **Enterprise Information Systems**, Taylor & Francis, v. 14, n. 8, p. 1150–1171, 2020.

LI, En; ZENG, Liekang; ZHOU, Zhi; CHEN, Xu. Edge ai: On-demand accelerating deep neural network inference via edge computing. **IEEE Transactions on Wireless Communications**, IEEE, v. 19, n. 1, p. 447–457, 2019.

LI, Wei; SHI, Yahong; LI, Yajie. Research on secure control and communication for cyber-physical systems under cyber-attacks. **Transactions of the Institute of Measurement and Control**, SAGE Publications Sage UK: London, England, v. 41, n. 12, p. 3421–3437, 2019.

LI, Zhongwen; ZANG, Chuanzhi; ZENG, Peng; YU, Haibin; LI, Hepeng. Mas based distributed automatic generation control for cyber-physical microgrid system. **IEEE/CAA Journal of Automatica Sinica**, IEEE, v. 3, n. 1, p. 78–89, 2016.

LIANG, Limei; LIU, Shuai; XU, Haotian; SU, Rong; LI, Yueyang. Distributed cyber attack detection and physical fault diagnosis for a class of interconnected large-scale systems. **ISA transactions**, Elsevier, v. 148, p. 182–190, 2024.

LIAO, Yongxin; DESCHAMPS, Fernando; LOURES, Eduardo de Freitas Rocha; RAMOS, Luiz Felipe Pierin. Past, present and future of industry 4.0-a systematic literature review and research agenda proposal. **International journal of production research**, Taylor & Francis, v. 55, n. 12, p. 3609–3629, 2017.

LIMEIRA, Marcelo; PIARDI, Luis; KALEMPA, Vivian Cremer; LEITÃO, Paulo; OLIVEIRA, André Schneider de. Depthlidar: Active segmentation of environment depth map into mobile sensors. **IEEE Sensors Journal**, IEEE, v. 21, n. 17, p. 19047–19057, 2021.

LIN, Shi-Wan; SIMMON, Eric; MELLOR, Stephen; MILLER, Bradford; DURAND, Jacques; BLEAKLEY, Graham; AMINE, Chigani; ROBERT, Martin; BRETT, Murphy; CRAWFORD, Mark. **The Industrial Internet of Things Volume G1: Reference Architecture**. 2019.

LOMBARDI, Marco; PASCALE, Francesco; SANTANIELLO, Domenico. Internet of things: A general overview between architectures, protocols and applications. **Information**, MDPI, v. 12, n. 2, p. 87, 2021.

LOPES, João Victor Bonella; VILLA-PARRA, Ana Cecilia; BASTOS-FILHO, Teodiano. A cyber-physical system for low cost monitoring and sensing of rural areas using sensors, microcontrollers and lora network: Agriculture 4.0. *In*: SPRINGER. **Proceedings of the 3rd International Conference on Human Interaction and Emerging Technologies: Future Applications, Paris, France**. 2021. p. 461–467.

LU, Yang. Industry 4.0: A survey on technologies, applications and open research issues. **Journal of industrial information integration**, Elsevier, v. 6, p. 1–10, 2017.

MAFFEI, Alessio; SRINIVASAN, Seshadhri; MEOLA, Daniela; PALMIERI, Giovanni; IANNELLI, Luigi; HOLHJEM, Øystein Hov; MARAFIOTI, Giancarlo; MATHISEN, Geir; GLIELMO, Luigi. A cyber-physical systems approach for implementing the receding horizon optimal power flow in smart grids. **IEEE Transactions on Sustainable Computing**, IEEE, v. 3, n. 2, p. 98–111, 2017.

MAKANDA, Inno Lorren Désir; JIANG, Pingyu; YANG, Maolin; SHI, Haoliang. Emergence of collective intelligence in industrial cyber-physical-social systems for collaborative task allocation and defect detection. **Computers in Industry**, Elsevier, v. 152, p. 104006, 2023.

MELO, Mário; AQUINO, Gibeon. Fatema: A framework for multi-layer fault tolerance in iot systems. **Sensors**, MDPI, v. 21, n. 21, p. 7181, 2021.

MERABET, Ghezlane Halhoul; ESSAAIDI, Mohammed; TALEI, Hanaa; ABID, Mohamed Riduan; KHALIL, Nacer; MADKOUR, Mohcine; BENHADDOU, Driss. Applications of multi-agent systems in smart grids: A survey. *In*: IEEE. **2014 International conference on multimedia computing and systems (ICMCS)**. 2014. p. 1088–1094.

MISHRA, Ayaskanta; JHA, Amitkumar V; APPASANI, Bhargav; RAY, Arun Kumar; GUPTA, Deepak Kumar; GHAZALI, Abu Nasar. Emerging technologies and design aspects of next generation cyber physical system with a smart city application perspective. **International Journal of System Assurance Engineering and Management**, Springer, v. 14, n. Suppl 3, p. 699–721, 2023.

MONOSTORI, László; KÁDÁR, Botond; BAUERNHANSL, Thomas; KONDOH, Shinsuke; KUMARA, Soundar; REINHART, Gunther; SAUER, Olaf; SCHUH, Gunther; SIHN, Wilfried; UEDA, Kenichi. Cyber-physical systems in manufacturing. **Cirp Annals**, Elsevier, v. 65, n. 2, p. 621–641, 2016.

MORTELLEC, Antoine Le; CLARHAUT, Joffrey; SALLEZ, Yves; BERGER, Thierry; TRENTESAUX, Damien. Embedded holonic fault diagnosis of complex transportation systems. **Engineering Applications of Artificial Intelligence**, Elsevier, v. 26, n. 1, p. 227–240, 2013.

MOUFADDAL, Mariam; BENGHABRIT, Asmaa; BOUHADDOU, Imane. A cyber-physical warehouse management system architecture in an industry 4.0 context. *In*: SPRINGER. **International Conference on Artificial Intelligence & Industrial Applications**. 2020. p. 125–148.

MOUSSA, Noureddine; HAMIDI-ALAOUI, Zakaria; ALAOUI, Abdelbaki El Belrhiti El. Cftm: A centralized fault tolerant mechanism for wireless sensor networks. *In*: IEEE. **2019 5th International Conference on Optimization and Applications (ICOA)**. 2019. p. 1–6.

MUDASSAR, Muhammad; ZHAI, Yanlong; LIAO, Lejian; SHEN, Jun. A decentralized latency-aware task allocation and group formation approach with fault tolerance for iot applications. **IEEE Access**, IEEE, v. 8, p. 49212–49223, 2020.

MURATA, Tadao. Petri nets: Properties, analysis and applications. **Proceedings of the IEEE**, IEEE, v. 77, n. 4, p. 541–580, 1989.

NAZARENKO, Artem A; CAMARINHA-MATOS, Luis M. The role of digital twins in collaborative cyber-physical systems. *In*: SPRINGER. **Proceedings of Advanced Doctoral Conference on Computing, Electrical and Industrial Systems (DoCEIS 2020), Costa de Caparica, Portugal, 2020**. 2020. p. 191–205.

NGUYEN, Thang Trung; MOHAMMADI, Fazel. Cyber-physical power and energy systems with wireless sensor networks: A systematic review. **Journal of Electrical Engineering & Technology**, Springer, v. 18, n. 6, p. 4353–4365, 2023.

NICULESCU-MIZIL, Alexandru; CARUANA, Rich. Predicting good probabilities with supervised learning. *In*: **Proceedings of the 22nd international conference on Machine learning**. 2005. p. 625–632.

NIKOLAKIS, Nikolaos; MARATOS, Vasilis; MAKRIS, Sotiris. A cyber physical system (cps) approach for safe human-robot collaboration in a shared workplace. **Robotics and Computer-Integrated Manufacturing**, Elsevier, v. 56, p. 233–243, 2019.

ODELL, James J; PARUNAK, Harry Van Dyke; BAUER, Bernhard. Representing agent interaction protocols in uml. *In*: SPRINGER. **International Workshop on Agent-Oriented Software Engineering**. 2000. p. 121–140.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine Learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011.

PEFFERS, Ken; TUUNANEN, Tuure; ROTHENBERGER, Marcus A; CHATTERJEE, Samir. A design science research methodology for information systems research. **Journal of management information systems**, Taylor & Francis, v. 24, n. 3, p. 45–77, 2007.

PENG, Lianghong; CAO, Xianghui; SUN, Changyin. Optimal transmit power allocation for an energy-harvesting sensor in wireless cyber-physical systems. **IEEE Transactions on Cybernetics**, IEEE, v. 51, n. 2, p. 779–788, 2019.

PIARDI, Luis; COSTA, Pedro; OLIVEIRA, André; LEITÃO, Paulo. Collaborative Fault Detection and Diagnosis Architecture for Industrial Cyber-Physical Systems. *In*: **IEEE Int. Conf. on Industrial Technology (ICIT)**. 2022. p. 1–6.

PIARDI, Luis; COSTA, Pedro; OLIVEIRA, André; LEITÃO, Paulo. Mas-based distributed cyber-physical system in smart warehouse. **IFAC-PapersOnLine**, Elsevier, v. 56, n. 2, p. 6376–6381, 2023.

PIARDI, Luis; COSTA, Pedro; OLIVEIRA, André Schneider De; LEITÃO, Paulo. Collaborative fault tolerance for cyber-physical systems: The diagnosis stage. **IEEE Access**, IEEE, 2025.

PIARDI, Luis; de Oliveira, André Schneider; COSTA, Pedro; LEITãO, Paulo. Collaborative fault tolerance for cyber–physical systems: The detection stage. **Computers in Industry**, v. 166, p. 104253, 2025. ISSN 0166-3615.

PIARDI, Luis; KALEMPA, Vivian Cremer; LIMEIRA, Marcelo; OLIVEIRA, André Schneider de; LEITÃO, Paulo. Arena—augmented reality to enhanced experimentation in smart warehouses. **Sensors**, MDPI, v. 19, n. 19, p. 4308, 2019.

PIARDI, Luis; LEITÃO, Paulo; COSTA, Pedro; OLIVEIRA, André Schneider de. Collaboration and self-organization to enable self-healing in industrial cyber-physical systems. *In*:

SPRINGER. **International Workshop on Service Orientation in Holonic and Multi-Agent Manufacturing**. 2023. p. 532–543.

PIARDI, Luis; LEITÃO, Paulo; OLIVEIRA, André Schneider de. Fault-tolerance in cyber-physical systems: literature review and challenges. *In*: IEEE. **2020 IEEE 18th International Conference on Industrial Informatics (INDIN)**. 2020. v. 1, p. 29–34.

PIARDI, Luis; LEITAO, Paulo; QUEIROZ, Jonas; PONTES, Joseane. Role of digital technologies to enhance the human integration in industrial cyber–physical systems. **Annual Reviews in Control**, Elsevier, v. 57, p. 100934, 2024.

PIARDI, Luis; OLIVEIRA, André; COSTA, Pedro; LEITÃO, Paulo. Evaluation metrics for collaborative fault detection and diagnosis in cyber-physical systems. *In*: IEEE. **2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA)**. 2024. p. 1–8.

PIVOTO, Diego GS; ALMEIDA, Luiz FF De; RIGHI, Rodrigo da Rosa; RODRIGUES, Joel JPC; LUGLI, Alexandre Baratella; ALBERTI, Antonio M. Cyber-physical systems architectures for industrial internet of things applications in industry 4.0: A literature review. **Journal of manufacturing systems**, Elsevier, v. 58, p. 176–192, 2021.

PULLUM, Laura L. **Software fault tolerance techniques and implementation**. : Artech House, 2001.

PUNDIR, Amit; SINGH, Sanjeev; KUMAR, Manish; BAFILA, Anil; SAXENA, Geetika J. Cyber-physical systems enabled transport networks in smart cities: Challenges and enabling technologies of the new mobility era. **IEEe Access**, IEEE, v. 10, p. 16350–16364, 2022.

QI, Qinglin; ZHAO, Dongming; LIAO, T Warren; TAO, Fei. Modeling of cyber-physical systems and digital twin based on edge computing, fog computing and cloud computing towards smart manufacturing. *In*: AMERICAN SOCIETY OF MECHANICAL ENGINEERS. **International manufacturing science and engineering conference**. 2018. v. 51357, p. V001T05A018.

QIN, Jiahu; MA, Qichao; SHI, Yang; WANG, Long. Recent advances in consensus of multi-agent systems: A brief survey. **IEEE Transactions on Industrial Electronics**, IEEE, v. 64, n. 6, p. 4972–4983, 2016.

QUEIROZ, Jonas; LEITÃO, Paulo; BARBOSA, José; OLIVEIRA, Eugénio. Distributing intelligence among cloud, fog and edge in industrial cyber-physical systems. *In*: **16th International Conference on Informatics in Control, Automation and Robotics, ICINCO 2019**. 2019. p. 447–454.

QUEIROZ, Jonas; LEITÃO, Paulo; BARBOSA, José; OLIVEIRA, Eugénio; GARCIA, Gisela. Agent-based distributed data analysis in industrial cyber-physical systems. **IEEE Journal of Emerging and Selected Topics in Industrial Electronics**, IEEE, v. 3, n. 1, p. 5–12, 2021.

RAJKUMAR, Ragunathan. A cyber–physical future. **Proceedings of the IEEE**, IEEE, v. 100, n. Special Centennial Issue, p. 1309–1312, 2012.

RAMUDU, Kama; BANSAL, Sushil Kumar; SHAHNAZEER, CK; YASEEN, Syed Mufassir; ALIMOVNA, Pardayeva Zulfizar; ARUMUGAM, Mahendran. 6g cyber physical system based smart healthcare modelling by mobile edge network and artificial intelligence. **Wireless Personal Communications**, Springer, p. 1–16, 2024.

REHMAN, AU; AGUIAR, Rui L; BARRACA, Joao Paulo. Fault-tolerance in the scope of software-defined networking (sdn). **IEEE access**, IEEE, v. 7, p. 124474–124490, 2019.

REIJNEN, Ferdie; LELIVELD, Eva-Britt; MORTEL-FRONCZAK, Joanna van de; DINTHER, John van; ROODA, Jacobus; FOKKINK, Wan. Synthesized fault-tolerant supervisory controllers, with an application to a rotating bridge. **Computers in Industry**, Elsevier, v. 130, p. 103473, 2021.

REN, Lei; ZHANG, Lin; WANG, Lihui; TAO, Fei; CHAI, Xudong. Cloud manufacturing: key characteristics and applications. **International journal of computer integrated manufacturing**, Taylor & Francis, v. 30, n. 6, p. 501–515, 2017.

REPPA, Vasso; POLYCARPOU, Marios M; PANAYIOTOU, Christos G. Distributed sensor fault diagnosis for a network of interconnected cyberphysical systems. **IEEE Transactions on Control of Network Systems**, IEEE, v. 2, n. 1, p. 11–23, 2014.

RIBEIRO, Luis; BARATA, Jose. Re-thinking diagnosis for future automation systems: An analysis of current diagnostic practices and their applicability in emerging it based production paradigms. **Computers in Industry**, Elsevier, v. 62, n. 7, p. 639–659, 2011.

RIJSWIJK, Kelly; KLERKX, Laurens; BACCO, Manlio; BARTOLINI, Fabio; BULTEN, Ellen; DEBRUYNE, Lies; DESSEIN, Joost; SCOTTI, Ivano; BRUNORI, Gianluca. Digital transformation of agriculture and rural areas: A socio-cyber-physical system framework to support responsibilisation. **Journal of Rural Studies**, Elsevier, v. 85, p. 79–90, 2021.

RODRIGUES, Nelson; OLIVEIRA, Eugenio; LEITÃO, Paulo. Decentralized and on-the-fly agent-based service reconfiguration in manufacturing systems. **Computers in Industry**, Elsevier, v. 101, p. 81–90, 2018.

ROJAS, Félix; JEREZ, Cristóbal; HACKL, Christoph M; KALMBACH, Oliver; PEREDA, Javier; LILLO, Jonathan. Faults in modular multilevel cascade converters–part ii: Fault

tolerance, fault detection and diagnosis, and system reconfiguration. **IEEE Open Journal of the Industrial Electronics Society**, IEEE, v. 3, p. 594–614, 2022.

RUMBAUGH, James. **The unified modeling language reference manual**. : Pearson Education India, 2005.

RUSSELL, Peter Norvig Stuart. **Artificial intelligence: a modern approach**. : Upper Saddle River, NJ, USA: Prentice Hall Press., 2010. v. 3 rd. ed.

SAFARI, Sepideh; ANSARI, Mohsen; KHDR, Heba; GOHARI-NAZARI, Pourya; YARI-KARIN, Sina; YEGANEH-KHAKSAR, Amir; HESSABI, Shaahin; EJLALI, Alireza; HENKEL, Jörg. A survey of fault-tolerance techniques for embedded systems from the perspective of power, energy, and thermal issues. **IEEE Access**, IEEE, p. 12229–12251, 2022.

SARTORI, Isabel; AMARO, Carolina Azerêdo; JÚNIOR, Maurício Bezerra de Souza; EMBIRUÇU, Marcelo. Detecção, diagnóstico e correção de falhas: Uma proposição consistente de definições e terminologias. **Ciência & Engenharia**, v. 21, p. 29–34, 2012.

SHAO, Siyu; YAN, Ruqiang; LU, Yadong; WANG, Peng; GAO, Robert X. Dcnn-based multi-signal induction motor fault diagnosis. **IEEE Transactions on Instrumentation and Measurement**, IEEE, v. 69, n. 6, p. 2658–2669, 2019.

SHI, Weisong; CAO, Jie; ZHANG, Quan; LI, Youhuizi; XU, Lanyu. Edge computing: Vision and challenges. **IEEE internet of things journal**, IEEE, v. 3, n. 5, p. 637–646, 2016.

SILVA, Manuel; VALETTE, Robert. Petri nets and flexible manufacturing. *In*: SPRINGER. **European Workshop on Applications and Theory in Petri Nets**. 1988. p. 374–417.

SINGH, Munindar P. Agent communication languages: Rethinking the principles. **Computer**, IEEE, v. 31, n. 12, p. 40–47, 1998.

SINGH, Upendra; DUA, Amit; KUMAR, Neeraj; GUIZANI, Mohsen. Qos aware uplink scheduling for m2m communication in lte/lte-a network: a game theoretic approach. **IEEE Transactions on Vehicular Technology**, IEEE, v. 71, n. 4, p. 4156–4170, 2021.

SKLIROS, Christos; MIGUEZ, Manuel Esperon; FAKHRE, Ali; JENNIONS, Ian. A review of model based and data driven methods targeting hardware systems diagnostics. PTDT, 2018.

SUZUKI, Ichiro; MURATA, Tadao. A method for stepwise refinement and abstraction of petri nets. **Journal of computer and system sciences**, Elsevier, v. 27, n. 1, p. 51–76, 1983.

TAN, Yifei; YANG, Wenhe; YOSHIDA, Kohtaroh; TAKAKUWA, Soemon. Application of iot-aided simulation to manufacturing systems in cyber-physical system. **Machines**, Multidisciplinary Digital Publishing Institute, v. 7, n. 1, p. 2, 2019.

TERTYTCHNY, Georgios; MICHAEL, Maria K. dataset reduction framework for intelligent fault detection in iot-based cyber-physical systems using machine learning techniques. *In*: IEEE. **2020 International Conference on Omni-layer Intelligent Systems (COINS)**. 2020. p. 1–6.

TRICOMI, Giuseppe; GIACOBBE, Maurizio; FICILI, Ilenia; PEDITTO, Nicola; PULIAFITO, Antonio. Smart city as cooperating smart areas: On the way of symbiotic cyber–physical systems environment. **Sensors**, MDPI, v. 24, n. 10, p. 3108, 2024.

UBAIDILLAHA, Sharifah Hafizah Sy Ahmad; NORAZIAHA, A; SAHABUDIN, N Aa. A survey on potential reactive fault tolerance approach for distributed systems in big data. *In*: **Proc. of SPIE Vol**. 2023. v. 12590, p. 1259009–1.

VALETTE, Robert. Analysis of petri nets by stepwise refinements. **Journal of computer and system sciences**, Elsevier, v. 18, n. 1, p. 35–46, 1979.

VAN, Mien; CEGLAREK, Dariusz. Robust fault tolerant control of robot manipulators with global fixed-time convergence. **Journal of the Franklin Institute**, Elsevier, v. 358, n. 1, p. 699–722, 2021.

VOROTNIKOV, Sergey; ERMISHIN, Konstantin; NAZAROVA, Anaid; YUSCHENKO, Arkady. Multi-agent robotic systems in collaborative robotics. *In*: SPRINGER. **International Conference on Interactive Collaborative Robotics**. 2018. p. 270–279.

WALKER, Paul. **RELX 2024 Annual Report**. 2024. Available on: https://www.relx.com/~/media/Files/R/RELX-Group/documents/reports/annual-reports/relx-2024-annual-report.pdf. Accessed: 05 jan. 2025.

WANG, Jinjiang; MA, Yulin; ZHANG, Laibin; GAO, Robert X; WU, Dazhong. Deep learning for smart manufacturing: Methods and applications. **Journal of manufacturing systems**, Elsevier, v. 48, p. 144–156, 2018.

WANG, Kaixuan; QIU, Xuesong; GUO, Shaoyong; QI, Feng. Fault tolerance oriented sensors relay monitoring mechanism for overhead transmission line in smart grid. **IEEE Sensors Journal**, IEEE, v. 15, n. 3, p. 1982–1991, 2014.

WANG, Shiyong; WAN, Jiafu; ZHANG, Daqiang; LI, Di; ZHANG, Chunhua. Towards smart factory for industry 4.0: a self-organized multi-agent system with big data based feedback and coordination. **Computer networks**, Elsevier, v. 101, p. 158–168, 2016.

WEISER, Christian; OSSMANN, Daniel. Fault-tolerant control for a high altitude long endurance aircraft. **IFAC-PapersOnLine**, Elsevier, v. 55, n. 6, p. 724–729, 2022.

WEISS, Gerhard. A modern approach to distributed artificial intelligence. **IEEE transactions on systems man & cybernetics-part c applications & reviews**, v. 22, n. 2, 1999.

WOOLDRIDGE, Michael. **An introduction to multiagent systems**. : John wiley & sons, 2009.

WOOLDRIDGE, Michael; CIANCARINI, Paolo. Agent-oriented software engineering: The state of the art. *In*: SPRINGER. **International workshop on agent-oriented software engineering**. 2000. p. 1–28.

WOOLDRIDGE, Michael J; JENNINGS, Nicholas R. Intelligent agents: Theory and practice. **The knowledge engineering review**, v. 10, n. 2, p. 115–152, 1995.

WUEST, Thorsten; WEIMER, Daniel; IRGENS, Christopher; THOBEN, Klaus-Dieter. Machine learning in manufacturing: advantages, challenges, and applications. **Production & Manufacturing Research**, Taylor & Francis, v. 4, n. 1, p. 23–45, 2016.

XIA, Dan; JIANG, Chun; WAN, Jiafu; JIN, Jiong; LEUNG, Victor CM; MARTÍNEZ-GARCÍA, Miguel. Heterogeneous network access and fusion in smart factory: A survey. **ACM Computing Surveys**, ACM New York, NY, v. 55, n. 6, p. 1–31, 2022.

XING, Liudong; TANNOUS, Massarrah; VOKKARANE, Vinod M; WANG, Honggang; GUO, Jun. Reliability modeling of mesh storage area networks for internet of things. **IEEE Internet of Things Journal**, IEEE, v. 4, n. 6, p. 2047–2057, 2017.

XU, Dezhi; ZHU, Fanglai; ZHOU, Zepeng; YAN, Xinggang. Distributed fault detection and estimation in cyber–physical systems subject to actuator faults. **ISA transactions**, Elsevier, v. 104, p. 162–174, 2020.

XU, Hansong; YU, Wei; GRIFFITH, David; GOLMIE, Nada. A survey on industrial internet of things: A cyber-physical systems perspective. **IEEE access**, IEEE, v. 6, p. 78238–78259, 2018.

XU, Li Da; XU, Eric L; LI, Ling. Industry 4.0: state of the art and future trends. **International journal of production research**, Taylor & Francis, v. 56, n. 8, p. 2941–2962, 2018.

YANG, Zhe; HUANG, Yunwei; NAZEER, Faisal; ZI, Yanyang; VALENTINO, Gianluca; LI, Chuan; LONG, Jianyu; HUANG, Huiyu. A novel fault detection method for rotating machinery based on self-supervised contrastive representations. **Computers in Industry**, Elsevier, v. 147, p. 103878, 2023.

YOO, YoungJun. Data-driven fault detection process using correlation based clustering. **Computers in Industry**, Elsevier, v. 122, p. 103279, 2020.

ZHANG, Haibo; GE, Dandan; LIU, Jizhen; ZHANG, Yi. Multifunctional cyber-physical system testbed based on a source-grid combined scheduling control simulation system. **IET Generation, Transmission & Distribution**, Wiley Online Library, v. 11, n. 12, p. 3144–3151, 2017.

ZHANG, Jun; PAN, Lei; HAN, Qing-Long; CHEN, Chao; WEN, Sheng; XIANG, Yang. Deep learning based attack detection for cyber-physical system cybersecurity: A survey. **IEEE/CAA Journal of Automatica Sinica**, IEEE, v. 9, n. 3, p. 377–391, 2021.

ZHANG, Kunwu; SHI, Yang; KARNOUSKOS, Stamatis; SAUTER, Thilo; FANG, Huazhen; COLOMBO, Armando Walter. Advancements in industrial cyber-physical systems: An overview and perspectives. **IEEE Transactions on Industrial Informatics**, IEEE, v. 19, n. 1, p. 716–729, 2022.

ZHANG, Ling; CUI, Yuchen; XIONG, Zhi; LIU, Jianye; LAI, Jizhou; LV, Pin. Research on adaptive multi-source information fault-tolerant navigation method based on no-reference system diagnosis. **Sensors**, MDPI, v. 19, n. 13, p. 2911, 2019.

ZHANG, Yuanyuan; LIU, Jinkun; HE, Wei. Adaptive fault-tolerant control for a nonlinear flexible aircraft wing system. **Asian Journal of Control**, Wiley Online Library, v. 21, n. 5, p. 2340–2351, 2019.

ZHAO, Li; LI, Wei; LI, YaJie. Research on co-design of dual security control and communication for nonlinear cps with actuator fault and fdi attacks. **Measurement and control**, SAGE Publications Sage UK: London, England, v. 55, n. 7-8, p. 767–782, 2022.

ZHAO, Li; LI, Wei; LI, Yajie; HAN, Nani; ZHENG, Naiqin. Integrated security control for nonlinear cps with actuator fault and fdi attack: An active attack-tolerant approach. *In*: MDPI. **Actuators**. 2023. v. 12, n. 5, p. 216.

ZHAO, Zuoquan; WANG, Jiadong; CAO, Jiawei; GAO, Wenchao; REN, Qinyuan. A fault-tolerant architecture for mobile robot localization. *In*: IEEE. **2019 IEEE 15th International Conference on Control and Automation (ICCA)**. 2019. p. 584–589.

ZHOU, Hao; TANG, Yu Hua; JIANG, Jing Fei. Design and implementation of scalable autonomous centralized fault-tolerant scheme for satellite system. **Advanced Materials Research**, Trans Tech Publ, v. 532, p. 813–817, 2012.

ZHOU, Keliang; LIU, Taigang; ZHOU, Lifeng. Industry 4.0: Towards future industrial opportunities and challenges. *In*: IEEE. **12th International conference on fuzzy systems and knowledge discovery (FSKD)**. 2015. p. 2147–2152.

ZIDI, Salah; MOULAHI, Tarek; ALAYA, Bechir. Fault detection in wireless sensor networks through SVM classifier. **IEEE Sensors Journal**, IEEE, v. 18, n. 1, p. 340–347, 2017.

APPENDIX

APPENDIX A – PUBLISHED PAPERS AND AWARDS

Published papers in the research field of the thesis work as 1º author:

Piardi, L., Oliveira, A., Costa, P., Leitão, P. (2025). Collaborative fault tolerance for cyber–physical systems: The detection stage. **Computers in Industry**, Vol. 166, 104253.

Piardi, L., Costa, P., Oliveira, A., Leitão, P. (2025). Collaborative fault tolerance for cyber–physical systems: The diagnosis stage. **IEEE ACCESS**, Vol. 13, (pp 37278-37289).

Piardi, L., Oliveira, A., Costa, P., Leitão, P. (2024, September). Evaluation Metrics for Collaborative Fault Detection and Diagnosis in Cyber-Physical Systems. In **2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA24)** (pp. 1-8). IEEE.

Piardi, L., Leitão, P., Costa, P., Schneider de Oliveira, A. (2023, September). Collaboration and Self-organization to Enable Self-healing in Industrial Cyber-Physical Systems. In **International Workshop on Service Orientation in Holonic and Multi-Agent Manufacturing (SOHOMA2023)** (pp. 532-543). Cham: Springer Nature Switzerland.

Piardi, L., Costa, P., Oliveira, A., Leitão, P. (2023). MAS-based Distributed Cyber-physical System in Smart Warehouse. In **IFAC World Congress 2023**, Yokohama, Japan, IFAC-PapersOnLine, 56(2), 6376-6381.

Piardi, L., Costa, P., Oliveira, A., Leitão, P. (2022, August). Collaborative fault detection and diagnosis architecture for industrial cyber-physical systems. In **2022 IEEE International Conference on Industrial Technology (ICIT 2022)** (pp. 1-6). IEEE. ****Conference Best Student Paper Award Received****

Piardi, L., Leitão, P., Costa, P., de Oliveira, A. S. (2021, November). Fault-tolerance in cyber-physical systems using holonic multi-agent systems. In **International Workshop on Service Orientation in Holonic and Multi-Agent Manufacturing (SOHOMA2021)** (pp. 51-63). Cham: Springer International Publishing.

Piardi, L., Leitão, P., de Oliveira, A. S. (2020, July). Fault-tolerance in cyber-physical systems: literature review and challenges. In **2020 IEEE 18th International Conference on Industrial Informatics (INDIN2020)** (Vol. 1, pp. 29-34). IEEE.

Piardi, L., Kalempa, V. C., Limeira, M., de Oliveira, A. S., Leitão, P. (2019). Arena—augmented reality to enhanced experimentation in smart warehouses. **Sensors**, 19(19).

Published papers in the research field of the thesis work as co-author:

Kalempa, V. C., Piardi, L., Limeira, M., de Oliveira, A. S. (2023). Multi-Robot Task Scheduling for Consensus-Based Fault-Resilient Intelligent Behavior in Smart Factories. **Machines**, 11(4), 431.

Kalempa, V. C., Piardi, L., Limeira, M., de Oliveira, A. S. (2021). Multi-robot preemptive task scheduling with fault recovery: A novel approach to automatic logistics of smart factories. **Sensors**, 21(19), 6536.

Limeira, M., Piardi, L., Kalempa, V. C., Leitão, P., de Oliveira, A. S. (2021). Depthlidar: Active segmentation of environment depth map into mobile sensors. **IEEE Sensors Journal**, 21(17), 19047-19057.

Kalempa, V. C., Piardi, L., Limeira, M., de Oliveira, A. S. (2020). Fault-resilient collective ternary-hierarchical behavior to smart factories. **IEEE ACCESS**, Vol. 8, (pp 176905-176915).

Limeira, M., Piardi, L., Cremer Kalempa, V., Schneider, A., Leitão, P. (2020). Augmented reality system for multi-robot experimentation in warehouse logistics. In **Robot 2019: Fourth Iberian Robotics Conference: Advances in Robotics**, Vol. 1 (pp. 319-330). Springer International Publishing.

Limeira, M. A., Piardi, L., Kalempa, V. C., de Oliveira, A. S., Leitão, P. (2019, October). Wsbot: A tiny, low-cost swarm robot for experimentation on industry 4.0. In **2019 Latin American Robotics Symposium (LARS), 2019 Brazilian Symposium on Robotics (SBR) and 2019 Workshop on Robotics in Education (WRE)** (pp. 293-298). IEEE.

Contribution Organizing Special Session at Conferences:

Co-chair the Special Session *Mobile Robots for Industrial Environments* at the *Fourth Iberian Robotics Conference* (ROBOT2019) held in Porto from 20 to 22 November 2019.

Co-chair the Special Session *Mobile Robots for Industrial Environments* at the *Fifth Iberian Robotics Conference* (ROBOT2022) held in Porto from 23 to 25 November 2022.

Co-chair the Special Session *Advances and Optimization in Cyber-Physical Systems (AOCPS)* at the *International Conference on Optimization, Learning Algorithms and Applications* (OL2A2022) from 24 to 25 October 2022.

Figure 73 – Best student paper award received in IEEE ICIT 2022 conference.

Figure 74 – Highly commended paper award received in ROBOT2019 conference.

emerald PUBLISHING

Industrial Robot

International Journal of Robotics Research and Application

and

ROBOT 2019

Highly Commended Paper

Awarded to:

Marcelo Limeira, Luis Piardi, Vivian Kalempa, André Schneider and Paulo Leitão

For the Paper:

Augmented Reality System for Multi-Robot Experimentation in Warehouse Logistics

Clive Loughlin
Editor

Ciara Boardman
Publisher

APPENDIX B – MODULES OF CF* CONTROL LOOP LAYER

The *Data Acquisition*, *Data Processing & Control*, and *Data Visualization* modules represent the digitization of data used to execute local control of the physical component, sending commands to the set of actuators to reproduce the expected behavior. Depending on the asset that the agent wants to control, additional modules can be inserted or removed from the control loop layer, according to the application's needs. The function of these modules is to perform operational tasks related to monitoring and control, aiming to meet fast response requirements and ensure the operational autonomy of the component, i.e., without dependence on centralized monitoring and control node. Additionally, given their proximity to the physical components and their respective data sources, the control loop modules can continuously process and analyze the raw data to meet the control requirements of the asset. The control loop layer modules presented in CF* meta-model are described as:

The incorporation or removal of supplementary modules from the control loop layer is contingent upon the specific asset the agent intends to control, in alignment with the CPS application's requirements.

- ***Data acquisition***: this module manages the communication interfaces between the agent and the physical devices, ensuring the accurate digitization of sensor data. It plays a crucial role in cyber-physical components by enabling data collection from physical assets. This module can facilitate data acquisition for CPS-compatible components through standardized interfaces such as I2C or SPI, and analog or digital I/O ports. However, for legacy systems or older devices that were not initially designed for CPS integration, specialized adapters and handling of proprietary communication protocols may be required to interface with the system effectively. Once digitized, the raw data (ϵ_i) of $agent_i$ is made available to subsequent modules for further processing and control tasks. This modular approach ensures scalability and flexibility, allowing CPS to adapt to modern and legacy components while maintaining seamless operation within a distributed system.

- ***Data Visualization***: This module transforming raw data (ϵ_i) into comprehensible visual representations, enabling efficient integration, monitoring, and analysis of the component by users, operators, or maintenance teams through dashboards and Human Machine Interface (HMI). In other words, this module enhances the interpretability of the data by converting it into actionable information. It leverages advanced data visualization

techniques, such as bar charts, line charts, pie charts, heat maps, histograms, and interactive dashboards, promoting fast decision-making. An example of such visualization is the Grafana[1] platform, which is commonly used in CPS environments for presenting dynamic data insights. By employing these techniques, the module ensures that the vast amounts of sensor data generated by the agents in the CPS are not only accessible but also interpretable, promoting effective decision-making and system management.

- ***Data Processing & Control***: Upon receiving raw data ϵ_i as input, this module executes a pre-processing stage, including filtering, noise reduction, and data aggregation, preparing the information for subsequent processing and control tasks. For example, the pre-processing raw data in Lidar of AMR involves noise removal to ensure reliable sensor readings. The processing stage involves continuous analysis of the preprocessed data, which can range from simple rule-based operations to complex data-driven models. In the case of the AMR, the preprocessed Lidar data might be used to detect nearby objects (simple rule-based analysis) or be integrated into a Simultaneous Localization and Mapping (SLAM) algorithm to generate a map and determine the robot's position. Finally, the control defines the behavior of the physical component by executing operational logic. Continuing with the AMR example, the control module would process the robot's position data and generate commands for the actuators, enabling it to move and perform tasks based on its environment. This integrated flow from data acquisition to control ensures the system operates autonomously while meeting performance and response requirements.

Although the modules described in the control loop layer of the CF* meta-model operate autonomously to ensure the acquisition, processing, and data visualization necessary for the CPS operation, they also possess the potential for collaboration. This collaborative capability can be leveraged to perform distributed control tasks, optimize scheduling and task planning, and request specific services from other system components, ensuring more efficient resource management and coordinated execution of tasks. Note that while collaboration in the control loop layer enhances CPS performance, it is not the main focus of this work. The researches described in (LEITÃO; RESTIVO, 2006; BARBOSA *et al.*, 2015) and the practical applications presented in (PIARDI *et al.*, 2023) explore collaboration to enhance the operational control of CPS.

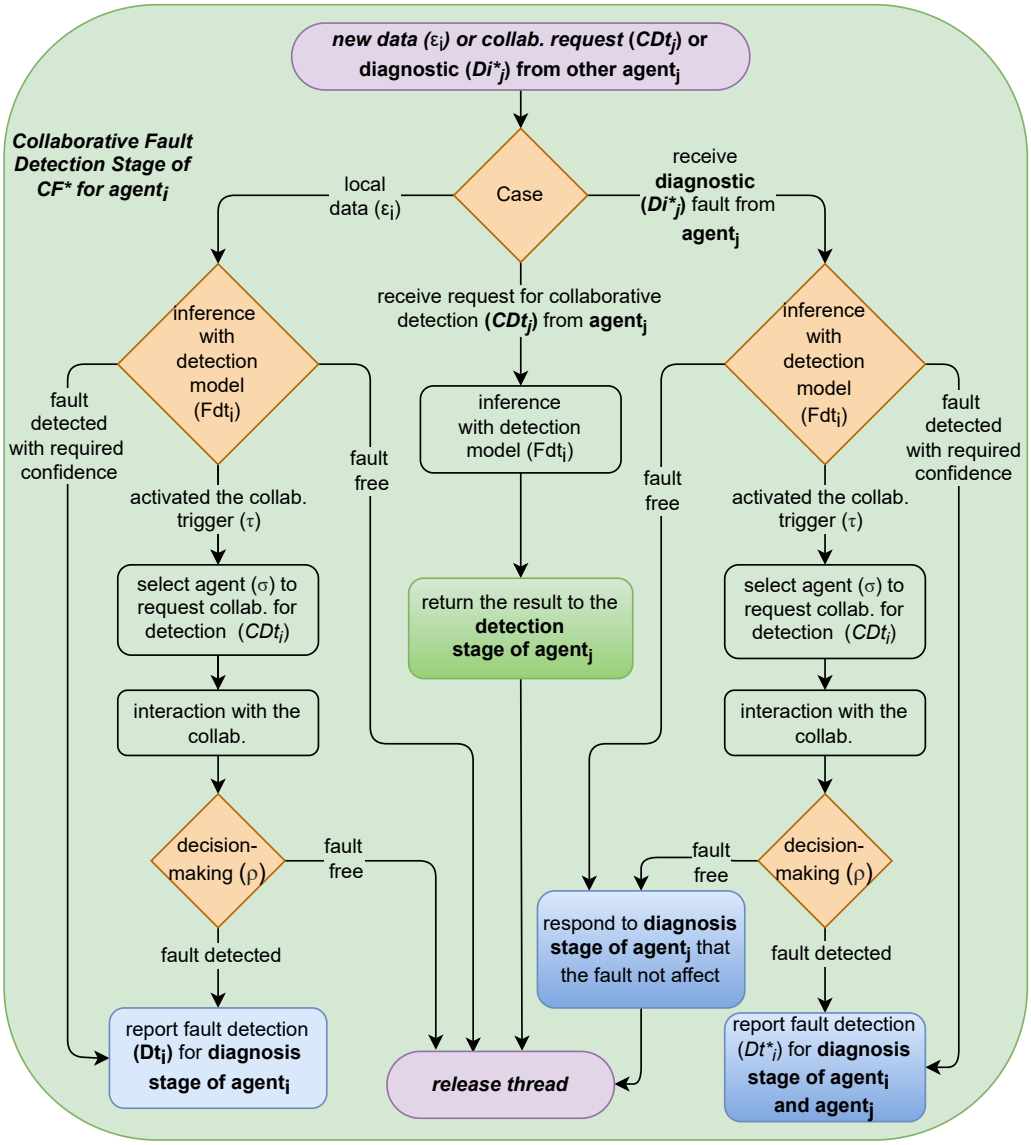
[1] Grafana is an open-source platform for monitoring and observability that allows users to visualize, query, and set alerts on metrics and logs from multiple sources https://grafana.com/)

APPENDIX C – FLOWCHARTS TO DESCRIBE THE CF* APPROACH

C.1 FLOWCHART OF COLLABORATIVE FAULT DETECTION

Figure 75 presents a flowchart detailing the internal static behavior of the fault detection module within the CF* approach. As shown in the flowchart, this stage can be triggered through three distinct pathways: either new local data (ϵ_i) becomes available for analysis to determine if a fault exists, or the $agent_i$ receives a fault detection collaboration request from an external agent ($agent_j$), or when the $agent_i$ receives an external fault diagnosis Di_i^* from the external $agent_j$. Each of these flows initiates the fault detection process, highlighting the flexibility of the system in both autonomous (with self-fault detection) and collaborative fault detection.

Figure 75 – Flowchart of internal operation for Collaborative Fault Detection of $agent_i$.



The flow that evolves in the left part of the flowchart in Figure 75 is activated when $agent_i$ receives new local data (ϵ_i) to analyze possible disturbances. In this scenario, the agent processes the data with its local fault detection model (Fdt_i), performing inferences to detect anomalies autonomously. This mechanism is crucial to ensure quick and efficient detection of faults that could compromise component performance. The information is promptly routed to the fault diagnosis stage if a fault is detected. However, the collaborative process will start if the collaboration trigger parameter (τ) is activated. In this case, the agent selects possible collaborators using the selection parameter (σ) and subsequently makes a decision (ρ) based on the interactions between the agents. If the fault is confirmed through collaboration, the diagnostic stage is triggered, otherwise, the fault tolerance processing thread is released.

The flow in the center of the flowchart in Figure 75 represents the scenario in which $agent_i$ receives a collaboration request from an external $agent_j$. Basically, the $agent_i$ analyzes the context sent by $agent_j$, based on its detection model (Fdt_i), and returns the contribution to $agent_j$. In this case, the model Fdt_i must be adapted to process the raw data from $agent_j$, as $Fdt_i(\epsilon_j) \rightarrow \{pdt_i, \varphi_j\}$.

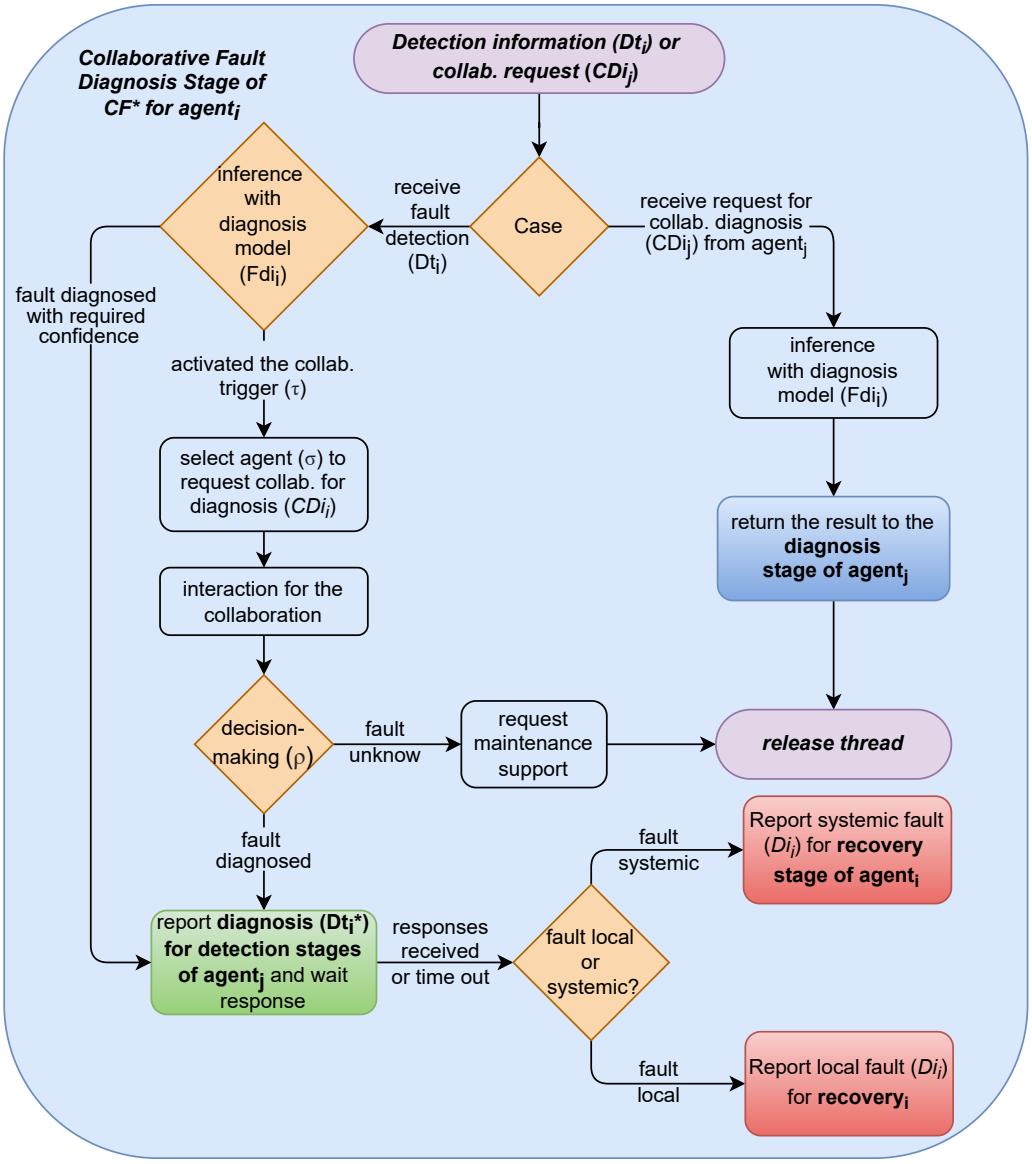
The flow on the right of the flowchart in Figure 75 represents the situation when $agent_i$ receives a diagnosis Di_j^* from an external $agent_j$, meaning that $agent_j$ has already detected and diagnosed a fault and is notifying potential agents that may be impacted to check if this fault also affects them. In this case, the Fdt_i model must be adjusted to process the Di_j^* diagnosis received. If the fault does not affect $agent_i$, it responds to $agent_j$, stating that the disturbance does not impact it. However, if the fault has an impact, $agent_i$ will communicate the impact to $agent_j$ and proceed with its diagnosis. At this point, it is up to $agent_j$ to assess the extent of the impact and take the lead in coordinating the recovery.

These three integrated flows provide a robust and scalable approach to the fault detection stage, combining local analysis with distributed collaboration, which enhance the precision and accuracy of detection in response to disturbances in the system. Additionally, during operation, $agent_i$ reports continuous feedback from the detection stage to $SystemModel_i$, which aims to adjust and improve both the detection model (Fdt_i) and the collaboration model (CDt_i). As needed, the $SystemModel_i$, can recommend updates to these models, ensuring that $agent_i$ remains adapted and effective in fault detection operations.

C.2 FLOWCHART OF COLLABORATIVE FAULT DIAGNOSIS

Figure 76 presents the flowchart detailing the internal behavior of the fault diagnosis module of the CF* approach. As illustrated in the flowchart, this stage can be initiated in two primary ways: either by receiving information from the detection stage within the same $agent_i$, or by receiving a request to collaborate in the fault diagnosis process from an external $agent_j$. These two pathways ensure flexibility and adaptability in fault diagnosis, allowing agents to operate autonomously while also supporting distributed fault diagnosis across the agents when necessary.

Figure 76 – Flowchart of internal operation for Collaborative Fault Diagnosis of $agent_i$.



The fault diagnosis flow, represented on the left side of the flowchart in Figure 76, is triggered when $agent_i$ receives information from the fault detection stage (Dt_i), indicating that

a fault has been detected with a confidence level of pdt_i, and the data associated with the fault is represented as φ_i. In this flow, the agent performs inference using its local fault diagnosis model (Fdi_i). The model then classifies the fault, identifying its cause and impact. Suppose the local model cannot confidently classify the fault or determine its cause and impact. In that case, the agent appeals to collaboration, according to the collaboration model CDi_i, interacting with compatible agents to enhance the diagnostic process.

Once the fault is diagnosed, a report is sent to the detection stages of potential external agents ($agent_j$) that may also be affected by the fault. After receiving feedback from these agents indicating whether or not they are impacted, the $agent_i$ evaluates whether the fault is local (i.e., only affects $agent_i$) or if it is global (affecting multiple agents within the system). The diagnostic output (Di_i) is then sent to the recovery stage of $agent_i$. If the fault is global and $agent_i$ was the first to detect it, $agent_i$ assumes responsibility for managing the recovery process. However, suppose the diagnosis is inconclusive in identifying the fault, its nature, or impact. In that case, the situation is reported for maintenance, and additional support is requested to manage the detected but undiagnosed fault.

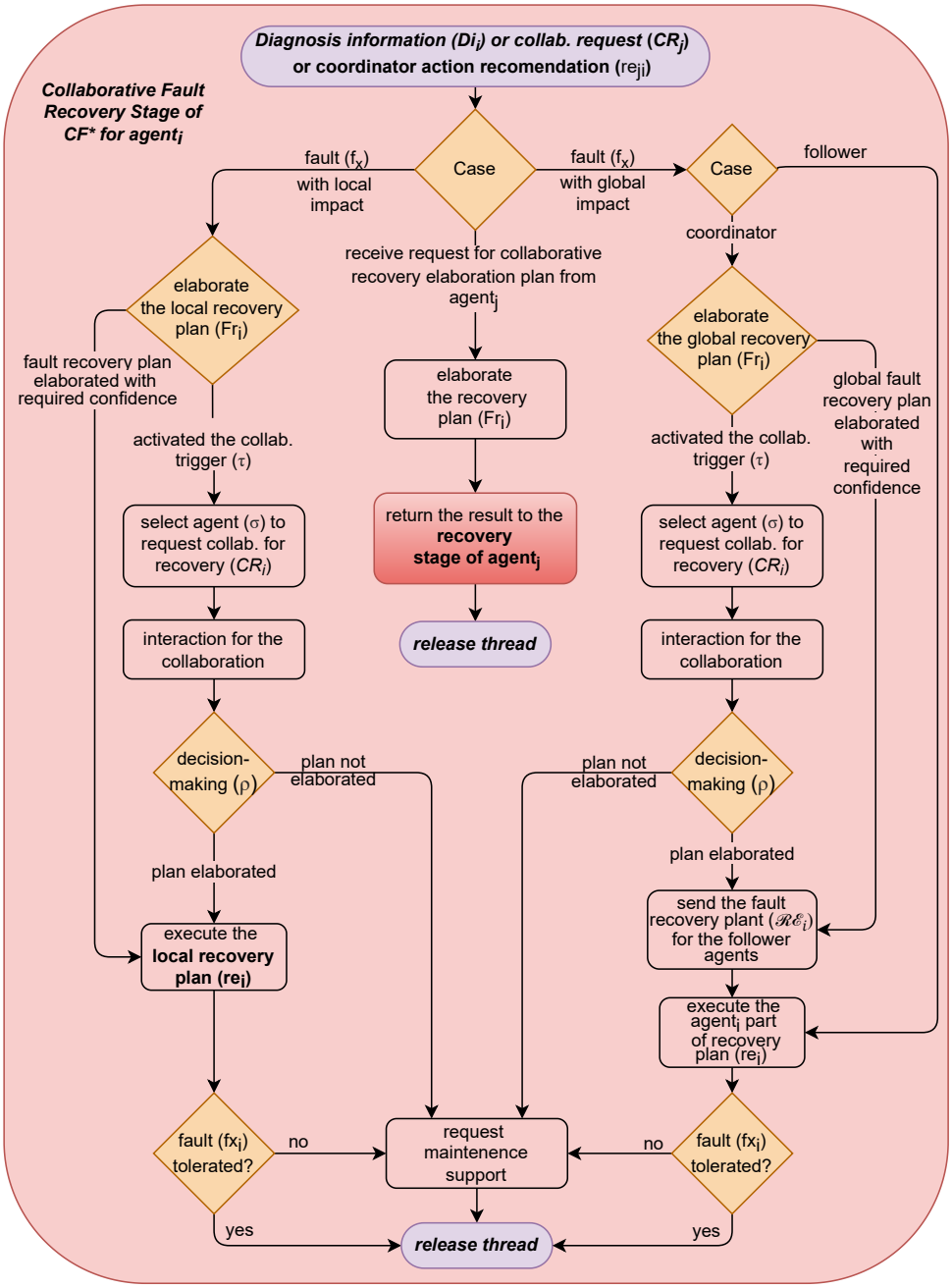
Alternatively, $agent_i$ can collaborate in diagnosing faults for $agent_j$, as represented by the flow on the right side of the flowchart in Figure 76. Like the detection stage, $agent_i$ may receive a collaboration request from an external $agent_j$ to assist in the fault diagnosis process. In this case, $agent_i$ analyzes the context and information provided by $agent_j$ using its own diagnostic model, Fdi_i, and offers feedback with its findings. This feedback aims to enhance $agent_j$ diagnostic capabilities, helping to improve the overall accuracy and robustness of the fault diagnosis.

The interaction and data exchange between distributed agents enhance fault diagnosis by providing a more holistic understanding of the CPS application and context. When an agent is unable to meet the required confidence levels using its local diagnostic model, collaboration extends its capacity by leveraging insights from the experiences and models of compatible agents within the system. Additionally, the diagnostic stage continuously adapts and refines its model during the operation of $agent_i$, with support from $SystemModel_i$, similar to the process in the fault detection stage.

C.3 FLOWCHART OF COLLABORATIVE FAULT RECOVERY

Figure 77 illustrates the flowchart detailing the internal static behavior of the fault recovery stage in the context of the CF*. As depicted in the flowchart, this stage can be triggered in three distinct ways: by receiving fault information $f(x)$ from the diagnosis stage within the same $agent_i$, by receiving a request to collaborate for the elaboration of recovery action plan from an external $agent_j$, or by receiving a recovery action recommendation from $agent_j$, where $agent_j$ assumes the role of coordinator and $agent_i$ takes on a follower role.

Figure 77 – Flowchart of internal operation for Collaborative Fault Recovery of $agent_i$.



The fault recovery flow represented on the left side of the flowchart in Figure 77 occurs when $agent_i$ receives information from the diagnosis stage (Di_i) reporting that the impact of the fault $\mathcal{IM}$ is local. In this scenario, by elaborating and executing the recovery actions plan locally within $agent_i$, the fault f_x will avoid propagating to other agents in the CPS. The fault recovery model Fr_i aims to devise actions that enable $agent_i$ to continue acceptably performing its functions, even in the presence of the fault. However, suppose the $agent_i$ is unable to generate appropriate recovery actions to mitigate or eliminate the fault's impact. In that case, it will engage in collaboration CR_i with other agents to formulate a suitable recovery action plan and then $agent_i$ will execute. Maintenance support will be requested if this collaborative action plan also proves inadequate to resolve the fault's impact or cannot be formulated.

The central flow of the recovery stage for $agent_i$ represents the possibility of collaboration in developing a recovery action plan for $agent_j$. Similar to the detection and diagnosis stages, $agent_i$ can receive a collaboration request from an external $agent_j$ to assist in the creation of a fault recovery action plan. In this scenario, $agent_i$ analyzes the context and information provided by $agent_j$, leveraging its own recovery model Fr_i, and provides feedback with its contribution to the recovery plan. This interaction enables the agents to share insights and solutions, enhancing the effectiveness of the recovery procedure.

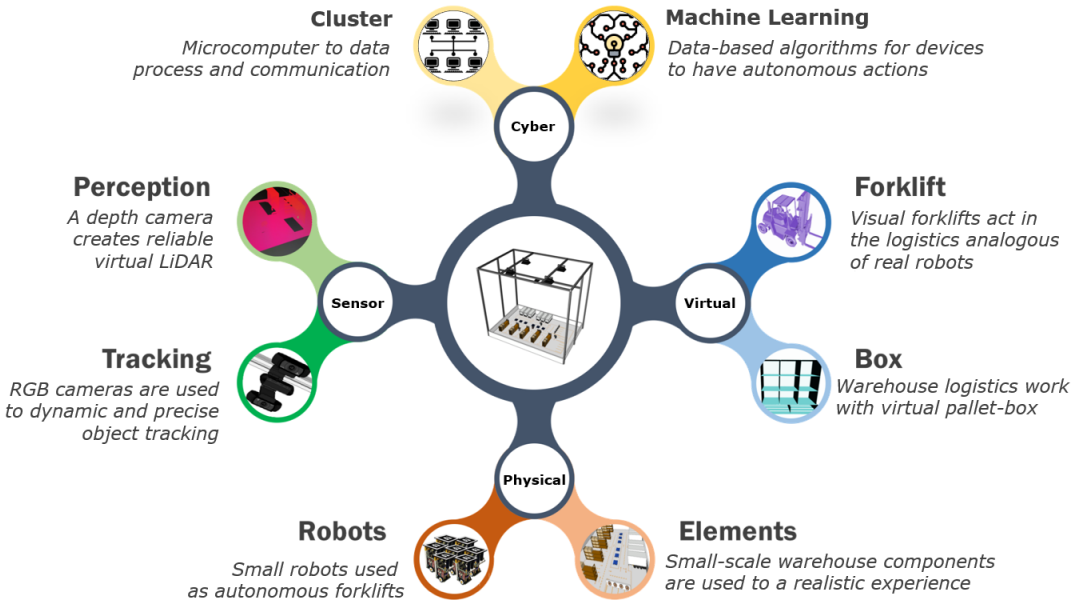
The flow represented on the right side of the flowchart in Figure 77 is triggered when $agent_i$ receives diagnostic information regarding a fault f_x with a global impact $\mathcal{IM}$, meaning the fault affects multiple agents. In this scenario, two situations may arise. In the first, the $agent_i$ was the first to diagnose the fault and is therefore responsible for coordinating the development of a recovery plan involving multiple agents. At this stage, $agent_i$ can devise the plan independently or request collaboration through CR_i to elaborate. Once the recovery set action plan ($\mathcal{RE}_i$) is finalized, the $agent_i$ will send the corresponding recovery actions to the respective agents involved, for example the re_ij is send to the $agent_j$.

Finally, in the second situation, the $agent_i$ may follow a recovery action plan devised by an external $agent_j$ (in this case, $agent_j$ was the first to diagnose the fault). Therefore, $agent_i$ will execute the recovery action re_{ji} formulated by $agent_j$, contributing to the system's collective intelligence to recover from a global fault collaboratively. Whether $agent_i$ acts as the coordinator or the follower, maintenance support will be requested if the recovery plan cannot be devised or executed.

APPENDIX D – DETAILS OF ARENA AND EXPERIMENTS FOR WAREHOUSE OPERATION

The ARENA is a workbench for active experimentation in logistic warehouses, developed by UTFPR and CeDRI research group, to replicate the real characteristics of a factory floor. The approach is specifically designed to evaluate novel methods that meet the requirements of I4.0, such as autonomous robotics for product transportation, IoT for data collection and system component monitoring, AI for data processing and analysis, AR for augmenting the real system with virtual elements, and CPS and MAS to integrate these technologies and warehouse components. Figure 78 presents the fundamental concepts supported be«y the ARENA workbench. The capabilities and infrastructure of the ARENA workbench were published in (PIARDI *et al.*, 2019).

Figure 78 – Fundamental concepts of ARENA.



The ARENA system is organized into four fundamental axes. The *Physical* axis corresponds to the logistic warehouse infrastructure, where real components, such as mobile robots, pallets, packages, and racks, are implemented at a reduced scale. A group of small mobile robots was developed to perform transport operations, replicating the behavior and principles of real forklifts.

The AR component introduces features through the inclusion of virtual elements in the *Virtual* axis. This allows the small robots to handle or transport product packages, simulating incoming and outgoing cargo operations. In this context, products are represented as virtual

elements that can be coupled with the mobile robots. New virtual components can also be added to the system to evaluate different scenarios, such as testing various types of virtual forklifts or analyzing human-robot interaction using virtual operators. Additionally, AR makes it possible to visually define warehouse zones, enhancing the observation and analysis of system behavior.

The *Sensor* axis refers to both global and local perception capabilities. The robots within the ARENA are localized using fiduciary markers detected by a global RGB camera sensor. A separate RGB-D camera generates a global point cloud, which is processed to extract LiDAR-like sensor data, enabling each tiny robot to be virtually equipped with LiDAR capabilities.

The *Cyber* axis is responsible for processing the large volume of data generated within ARENA. A cluster of low-cost microcomputers receives information from the physical devices and associated virtual products. These microcomputers can exchange information among themselves to optimize warehouse operations. Various algorithms based on AI and ML can be tested in the *Cyber* axis to improve logistics workflows, mobile robot path planning and task allocation, and FT functionalities.

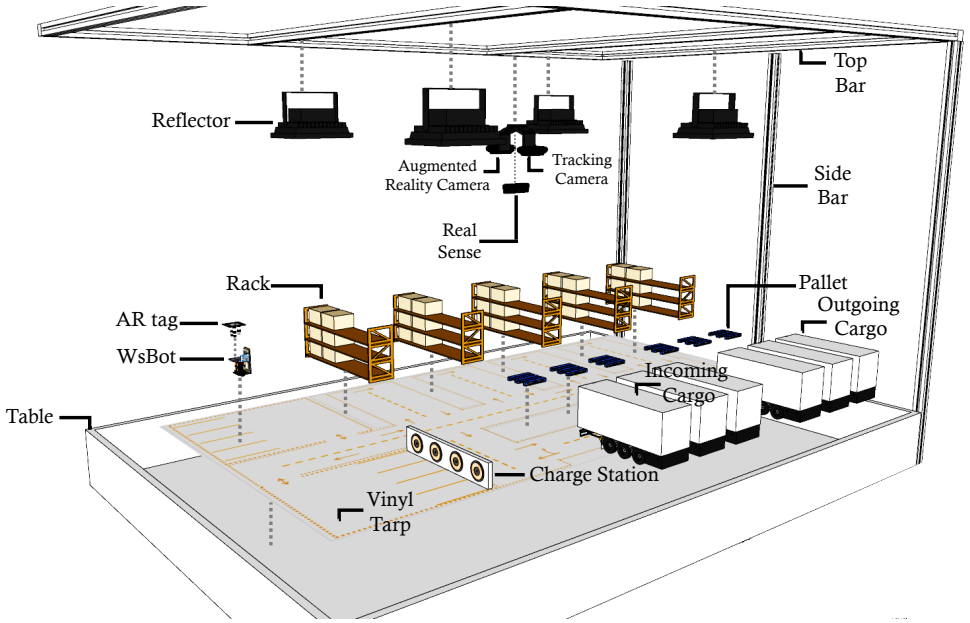
D.1 ARENA PHYSICAL STRUCTURE

The ARENA contains a physical structure that replicates the layout of a logistic warehouse at a reduced scale, preserving the essential characteristics and operational zones of a real warehouse. The ARENA is composed of a supporting table and components that simulate the warehouse floor and surroundings, including small mobile robots, AR elements, and virtual perception capabilities. This setup enables the representation of activities and interactions typical of a smart logistic warehouse. The infrastructure integrates all necessary elements to emulate the core functionalities of a warehouse system. The complete set of components is illustrated in Figure 79.

The infrastructure is mounted on a custom-built wooden table, measuring 1.68 meters in width and 1.04 meters in length, and supported by modular aluminum profiles. An aluminum frame, with a height of 1.6 meters, is attached to the table's sides and serves as a support for cameras and lighting equipment. On top of the table, a printed layout of the logistic warehouse is placed, depicting the robot track routes and operational sectors. This layout measures 1.48 meters in width by 0.84 meters in length, maintaining a 16:9 aspect ratio, and was printed on an A0-sized sheet.

An additional component integrated into ARENA is the charging station. It consists of

Figure 79 – Exploded view of the ARENA's infrastructure.



four wireless charging transmitters, each equipped with an electronic circuit and a coil to deliver energy to a corresponding receiving coil embedded in the robots. This setup allows the robots to recharge autonomously, eliminating the need for manual power connections or robots shutdowns.

To further enhance the perception of a real warehouse environment, scaled-down physical elements are included. These consist of wooden racks manufactured using Computer Numeric Control (CNC) machining, 3D-printed pallets, and cardboard boxes. All elements are designed in proportion to their real-world counterparts to ensure consistency and realism in the experimental setup.

D.2 RACKS INFRASTRUCTURE

Each rack in the ARENA environment is equipped with an associated IoT device responsible for digitizing temperature and humidity data. These measurements are essential for products that require specific storage conditions, such as perishable goods, which must be kept under low humidity to preserve shelf life, or pharmaceuticals, which require controlled low temperatures. The IoT rack device communicates temperature and humidity readings via Wi-Fi and consists of the following components:

1. *Development board:* A NodeMcu electronic development board;

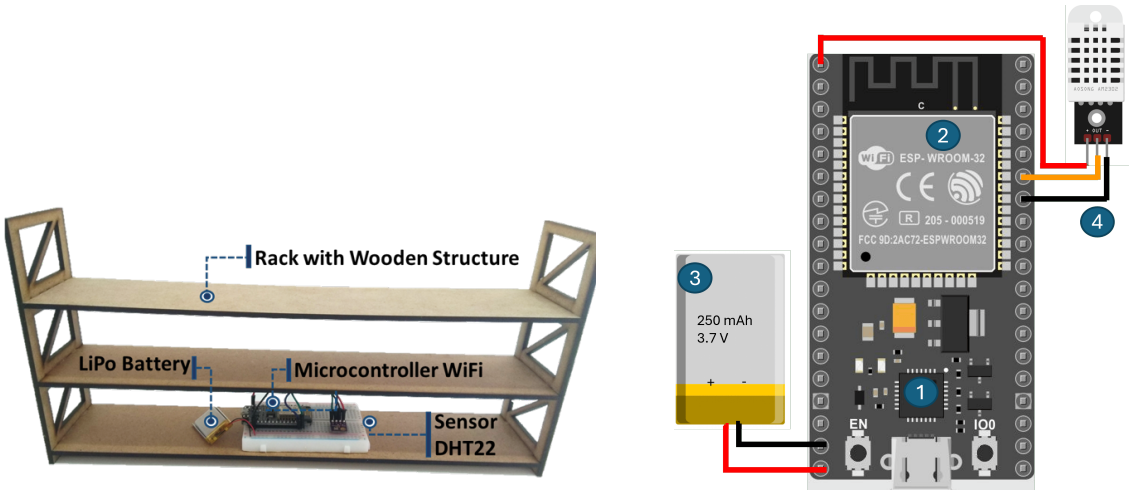
2. *WiFi microcontroller:* ESP32 microcontroller;

3. *Power supply:* 3.7 V Battery (also possible to connect direct to the DC power supply);

4. *Sensor:* DHT 22 with One-Wire communication Protocol;

To optimize battery usage, the device transmits data at configurable time intervals, considering that temperature and humidity typically vary slowly under standard warehouse conditions. In addition, the device includes digital and analog input/output ports, as illustrated in Figure 80, allowing for the integration of additional sensors during future developments.

Figure 80 – Rack infrastructure overview.



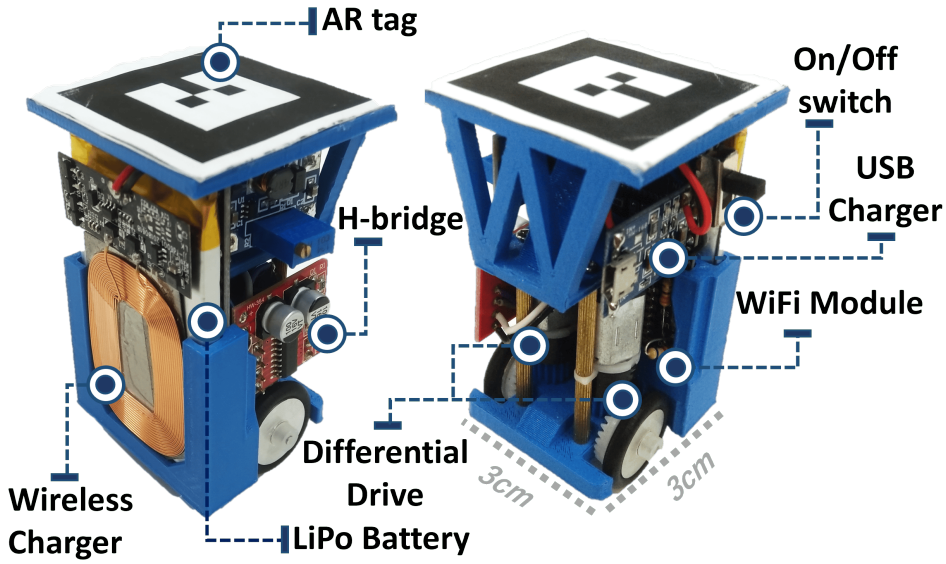
(a) Rack physical structure. **(b) Rack electronic device with DHT22 sensor.**

D.3 THE TINY AND LOW-COST MOBILE ROBOT: WSBOT

In ARENA, the forklift operations are performed by a small warehouse robot, referred to as the Warehouse Small roBot (WsBot), which interacts with other components to emulate the functions of a real warehouse forklift. The WsBot is a tiny mobile robot developed specifically for experimentation in small-scale warehouse environments, with the objective of evaluating intelligent collective behaviors in the context of I4.0. It is characterized by its low cost, ease of assembly, simplicity, and straightforward programmability. Figure 81 presents the WsBot.

These features make the WsBot particularly suitable for practical research in multi-robot systems. During its development, special attention was given to factors such as price, accessibility of components, and minimal complexity in assembly. The robot's actuation mechanism is inspired by real forklift systems and is based on a differential drive configuration. It is equipped with two wheels, each driven by a micro DC motor through a 90-degree gear connection. The

Figure 81 – WsBot description and components.



shared axis of the wheels is located slightly below the center of the robot's base. Each motor independently drives one of the wheels using a gear assembly, enabling control over linear and angular velocities by adjusting the supplied current and motor speed. Consequently, the angular velocity is proportional to the difference in power applied to the two motors. Instead of a conventional castor wheel, a 3 mm LED is used as a support point due to its small size, durability, and low friction coefficient. The mechanical specifications of the WsBot are summarized in Table 24.

Table 24 – Mechanical characteristics of WsBot.

Characteristic	Dimension	Unit
Width	33.0	mm
Length	33.0	mm
Height	70.0	mm
Wheel diameter	8.0	mm
Wheel thickness	2.5	mm
Mass	97	g

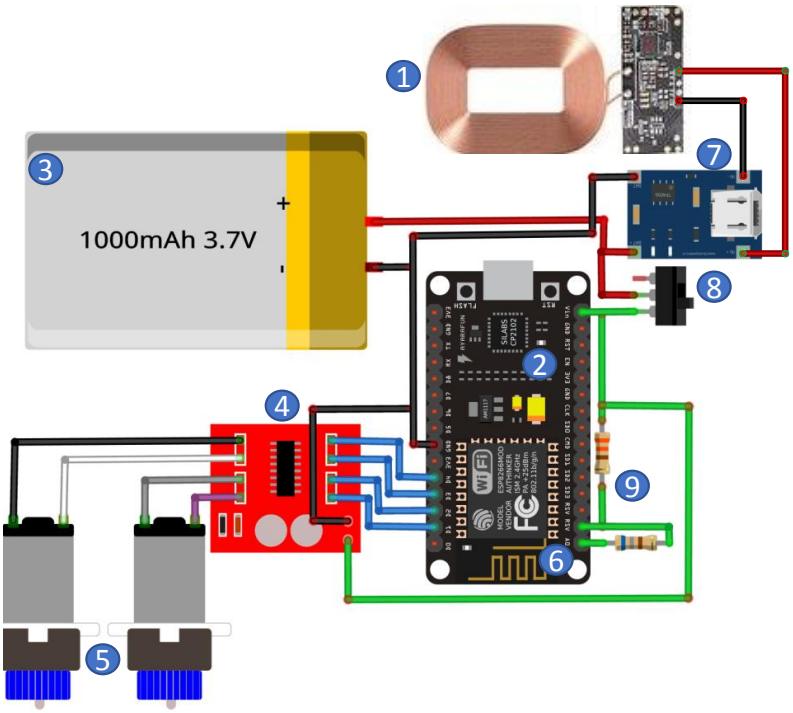
The mobile robot operates with three degrees of freedom, $[x, y, \theta]^T$, controlled via visual feedback. An RGB camera is dedicated to tracking the robots by identifying visual markers (i.e., AR-tags) attached to the top of each robot. Through image processing, the camera determines the position and orientation of each WsBot within the warehouse using unique AR-tag patterns. This enables the estimation of each robot's global pose.

The WsBot firmware is fully compatible with the ROS framework. Accordingly, velocity commands follow the standard $geometry_msg$ format, ensuring interoperability with other ROS-

compatible mobile robot systems. This feature allows the integration and testing of various robotic solutions within the ARENA environment.

The hardware design of the WsBot was projected to be quickly and ease of assembly and maintenance, facilitating the rapid production and replacement of units. The robot's electronic architecture consists of nine subsystems, as illustrated in Figure 82, each of which is described in detail in the following sections.

Figure 82 – WsBot electronic schematic.



1. *Wireless charger*: is a copper coil and a circuit board with a voltage regulator and power of $10W$;

2. *Development board*: electronic board to connect motors, H-bridge, switch,battery and sensors.

3. *Lithium Polymer (LiPo battery)*: is a rechargeable battery of $3.7V$ and $1100mAh$, size of 35x50x2mm and $25g$ of weight;

4. *H-bridge*: is the dual-driver power circuit to controls motors independently with a current of up to $1.5A$ for each engine;

5. *Differential drive*: is a set of small size DC motors that convert electrical impulses in mechanical movement, i.e., robot actuating;

6. *WiFi micro-controller:* is an ESP8266 microchip that manages the motors and devices connected in the development board. It also allows communication via WiFi with ROS server, enabling the cyber-physical device;

7. *USB charger*: is a battery charger capable of recharging without the need to remove it. It is powered by a mini-USB with a voltage of $5V$, providing a voltage of $4.2V$ for the battery, with a capacity of $1.2A$;

8. *On/off switch*: is a mechanical switch, with the function of turning the robot on and off. For safety reasons, it is advisable to charge the robot with it turned off;

9. *Voltage divider*: is the resistive circuit, with a coupling two resistors of 330Ω and 680Ω respectively, to adjust voltage scale to A/D converter that measure battery state.

D.4 SENSORS DEDICATED TO WSBOT IN ARENA

Most AMR platforms are equipped with embedded sensors that support localization functionalities, such as encoders, cameras, or LiDARs. However, the WsBot is a miniature robot that does not allow for the integration of such sensors due to its reduced size, limited payload, and constrained energy capacity. Another relevant factor is the cost added by these sensors to the overall price of each unit, especially considering that ARENA supports multiple robots operating simultaneously. In this context, an engineering solution was developed to provide localization and perception capabilities using an RGB and an RGB-D camera mounted above the ARENA environment, enabling the acquisition of robot poses and the simulation of virtual LiDAR sensors.

D.4.1 Localization System

The RGB camera is centrally mounted at the top of the supporting structure, offering a complete top-down view of the small-scale warehouse. Each WsBot has an AR-tag fixed to its upper surface, containing a geometric pattern with parameterized physical dimensions and a corresponding encoded ID. These markers are fully visible to the image acquisition system, enabling continuous visual tracking and localization of the WsBots within ARENA. Image processing techniques are employed to extract the robot's *pose* that is, its position (x,y,z) and

orientation θ in the Cartesian inertial frame $\mathbb{R}^3$. Since the AR-tag is aligned with the geometric center of the robot, the estimated position accurately corresponds to the robot's physical location.

To implement this functionality, the open-source library Alvar, developed by the VTT Technical Research Centre of Finland, was used in conjunction with the ROS package ar_track_alvar[1]. This package provides computer vision algorithms for identifying and tracking multiple distinct visual markers in a video stream.

During the development and experimentation phase in ARENA, the system was deployed on a Raspberry Pi 4 with a Quad-core processor @ 1.5GHz and 4 GB of RAM. Under this configuration, the system achieved a localization update rate of 16 Hz. Additionally, the (x, y) position accuracy was better than 1 cm, with acceptable variations in robot orientation estimates.

D.4.2 Virtual LiDAR

To implement the virtual LiDAR, the ar_track_alvar library is used to detect all visual markers and provide the *pose* of each AR-tag, which corresponds to the position of the respective mobile robot. The virtual sensor is then positioned precisely at the coordinates of the AR-tag, effectively placing it on top of each robot.

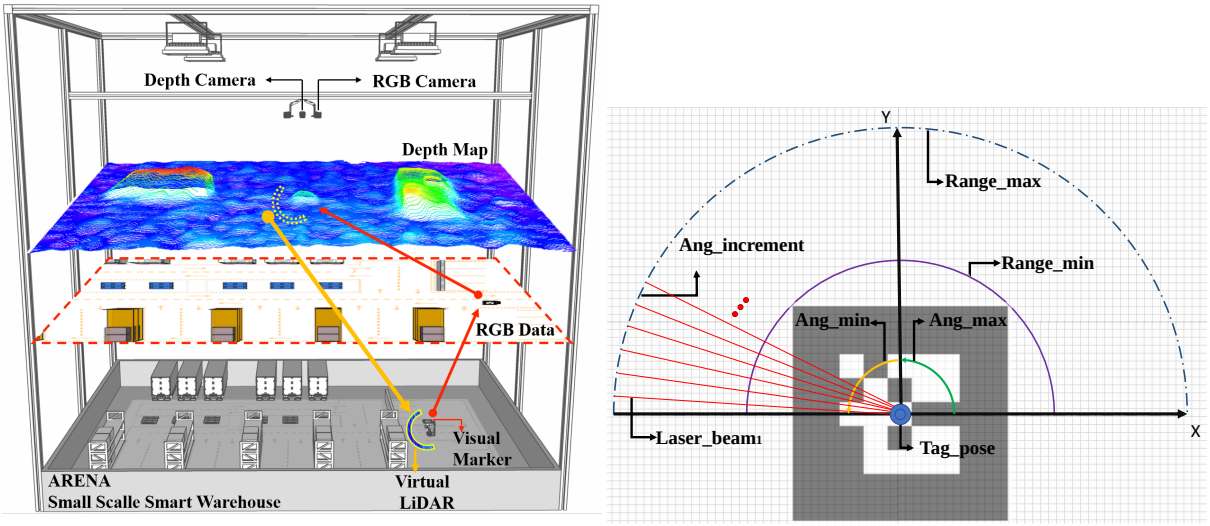
In summary, the virtual LiDAR generation process begins with acquiring the pose (x, y, z, θ) of each AR-tag—hence of each robot—through the RGB camera. Using this pose information, a data fusion process is carried out between the RGB and RGB-D cameras to localize the AR-tag within the point cloud. A pixel-by-pixel exploration is then performed in the around of each AR-tag's position within the point cloud to detect potential obstacles using depth data, as illustrated in Figure 83.

This method simulates a 2D laser scan around each detected AR-tag by systematically increasing the sensor's angular field and detection range. Whenever a depth value lower than the calibrated height of the ARENA floor is identified, the system infers the presence of an obstacle at that angle and distance. By aggregating this information across multiple directions and repeating the process for each robot pose, multiple virtual LiDAR scans are generated in parallel.

A comprehensive description of the procedures, parameters, and algorithms used to implement and validate the virtual LiDAR in ARENA is presented in detail in Limeira *et*

[1] https://wiki.ros.org/ar_track_alvar

Figure 83 – Virtual LiDAR overview.



(a) Elements to virtual LiDAR. **(b) Exploration process.**

al. (2021), developed in collaboration with other PhD students. The experimental validation employed an Intel RealSense D415 camera and a Hokuyo URG-04LX LiDAR as ground truth. The results demonstrated satisfactory precision. Since the camera is positioned at a considerable height above the ARENA and the environment lacks large vertical structures, shadowing effects caused by occlusions in the depth projection are negligible.

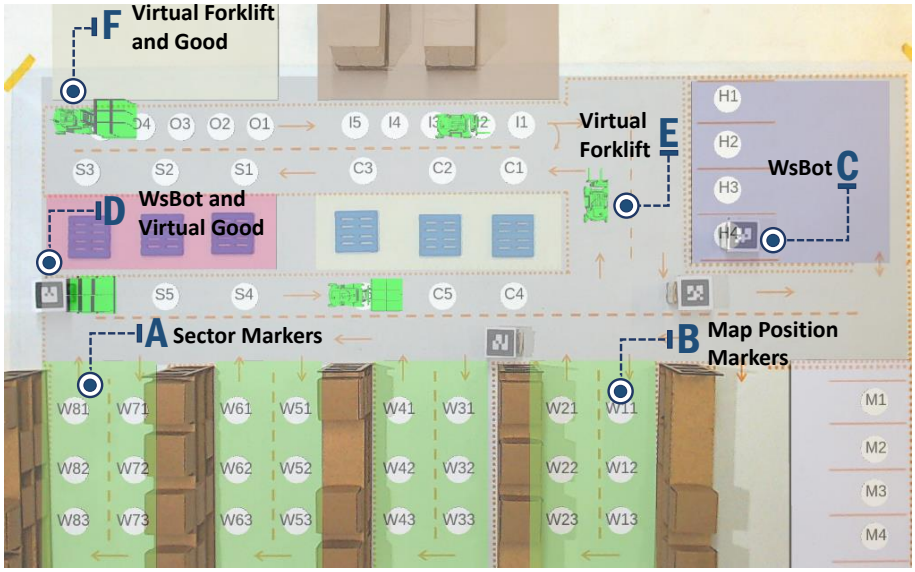
However, a notable drawback of this approach is the relatively low scan rate of the virtual LiDAR. Due to the extensive pixel-by-pixel exploration required, the method is computationally intensive, and its performance is directly limited by the available processing power.

D.5 AUGMENTED REALITY FEATURE

The camera dedicated to AR, fixed above ARENA and providing a top-down view, enables the superimposition of virtual layers and objects to create a more immersive and realistic experimentation experience. These virtual elements complement real warehouse operations. For example, the WsBot can transport virtual products from the incoming cargo area to the storage racks. This hybrid approach integrates virtual and physical components, allowing real-time interaction and the dynamic insertion of various graphical objects. Figure 84 illustrates the use of AR during an ARENA experiment.

The map position markers are used to define key locations within the warehouse, such as points for robot loading or for placing and retrieving goods from racks. Sector markers serve to visually distinguish warehouse zones—labeled “A” and “B”, for instance. These markers play

Figure 84 – Top view of the ARENA with AR element.



a crucial role in visualizing different operational areas and occupied positions, and they support the development of intelligent warehouse management strategies.

Dynamic components in the AR system change position as they perform tasks or interact with physical elements. For example, the element labeled “C” indicates a WsBot. While the robot itself is not a virtual entity, it becomes part of the AR context through its interaction with virtual components. One such interaction is illustrated by element “D”, where the robot transports a virtual product from one sector to another. This AR-based integration is particularly relevant for the current work, as the limited payload capacity of the robot restricts its ability to transport physical goods.

Although augmented reality is not the central focus of this workbench, which prioritizes the use of physical robots, the ARENA’s AR capabilities are worth highlighting. Elements “E” and “F” demonstrate the potential to incorporate other virtual objects, such as additional types of mobile robots like forklifts. While this feature will not be explored in the present work due to the sufficient number of physical robots available, it remains a valuable asset for future developments involving heterogeneous robot fleets.

The proposed hybrid system enables interaction between real and virtual elements, helping to reduce the intrinsic inaccuracies of purely simulated environments. Virtual-only systems often suffer from the so-called reality gap, in which models lack the fidelity required to accurately replicate real-world dynamics. ARENA addresses this challenge by combining augmented reality with the ROS framework, offering a more dynamic and realistic testing environment that remains grounded in real-world constraints. This approach leverages the speed,

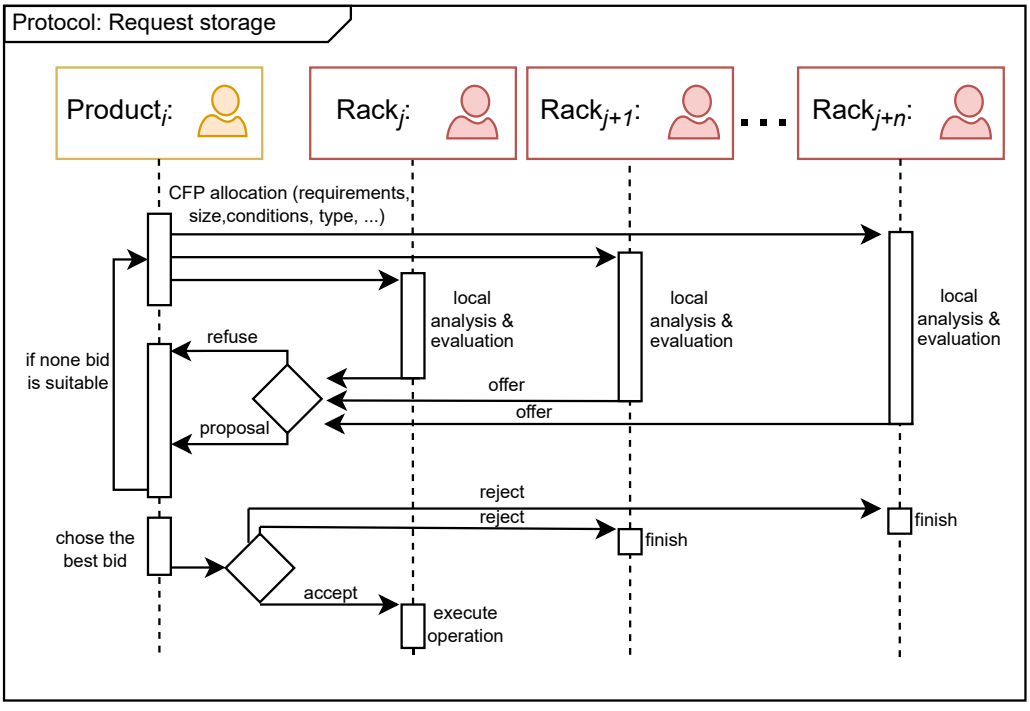
safety, and cost-effectiveness of simulation, while maintaining high fidelity through the use of real and virtual components—including both static and dynamic elements.

D.6 INTERACTION PROTOCOLS TO ARENA WAREHOUSE OPERATION

Have in mind that the a ARENA have a distributed proposal for warehouse operation, each agent, namely robot agent, rack agent and product agent makes its decision based on local information (obtained from the digitization of the physical component) complemented with information from the interaction and communication with the other agents.

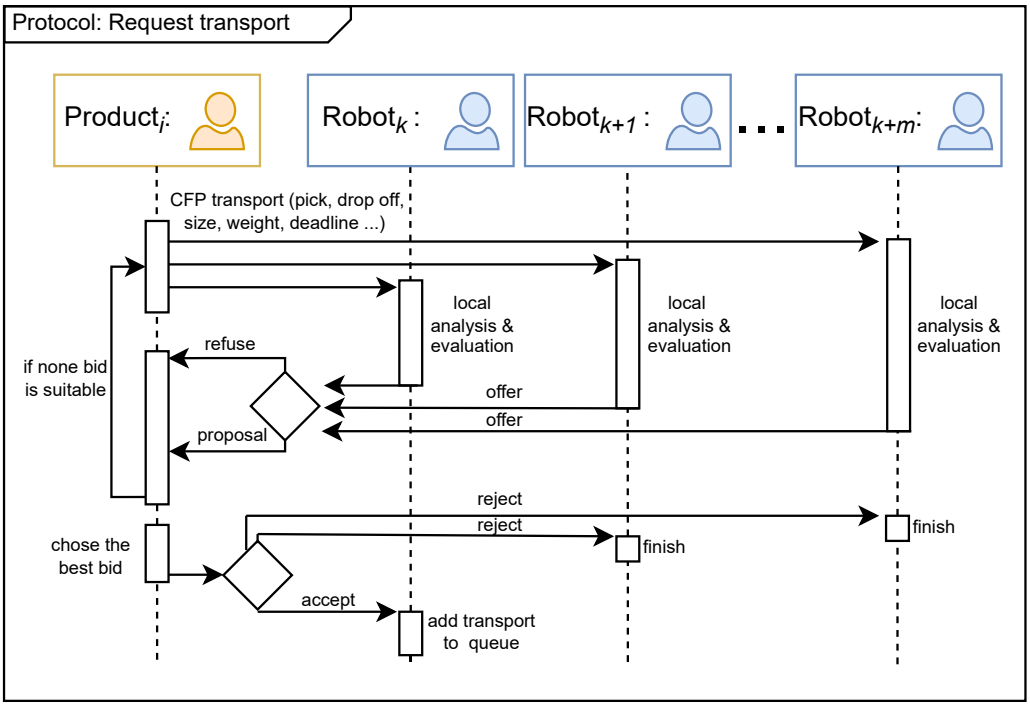
In the product agent process plan, the definition of areas that the product will pass through while in the warehouse directly impacts the quality and efficiency of the warehouse. In this sense, product agent negotiates the best space to allocate the physical part of the component. For this, a negotiation protocol formalized by UML is proposed in Fig. 85. The $productAgent_i$ agent (i indicates the index/id of the product agent) requires storage, and initiates Call for Proposal (CFP) describing storage conditions, their type and size for the n rack agent available in the warehouse. Based on the information from the CFP and the current status of each rack (i.e. free space, types of stored products) the $productAgent_i$ receives n offers from the rack agent. The $productAgent_i$ chooses the best offer and rejects the others. If no offer meets the needs of $productAgent_i$, a new CFP is started.

Figure 85 – Interaction for storage negotiation in the ARENA warehouse.



Similarly, a given $productAgent_i$ will need transport within the warehouse, represented in Fig. 86. Then, it performs a CFP informing m robot agent (m represent actives robots) pick and drop off location, size, weight, and deadline. Based on this information and the local conditions of the robot agent (distance, transport capacity, battery, number of routes), the $productAgent_i$ receives m offers from the TA. The $productAgent_i$ chooses the best offer and rejects the others.

Figure 86 – Interaction for transport negotiation in the ARENA warehouse.



This interaction between the different agents of the warehouses allows dynamic operation and quick adaptation in the face of a disturbance or insertion of new product agent. For both storage and transport processes, the appearance of a new batch of product agent or output/input of rack agent and robot agent in the system does not require executions of centralized optimization algorithms. Therefore, in a dynamic and complex scenario, such as a warehouse, each change results in an optimal solution and requires a new execution time. The proposed interaction aims at the system's dynamism, flexibility, and scalability in a tradeoff with optimization.

D.7 EXPERIMENTAL ANALYSIS FOR ARENA LOGISTIC OPERATION

In order to demonstrate the feasibility of the proposed architecture for a logistic warehouse, a series of experimental analyzes using the ARENA was conducted. First, the scalability of the system was analyzed, increasing the number of cyber agents robots. Then, a logistic warehouse operation scenario was established to analyze the transporting time to each product,

verifying the behavior when varying the number of robots or products representing a product arrival operation. Additionally, the divisibility of resources was investigated, verifying the amount of transport carried out by robots.

D.7.1 Scalability Analysis

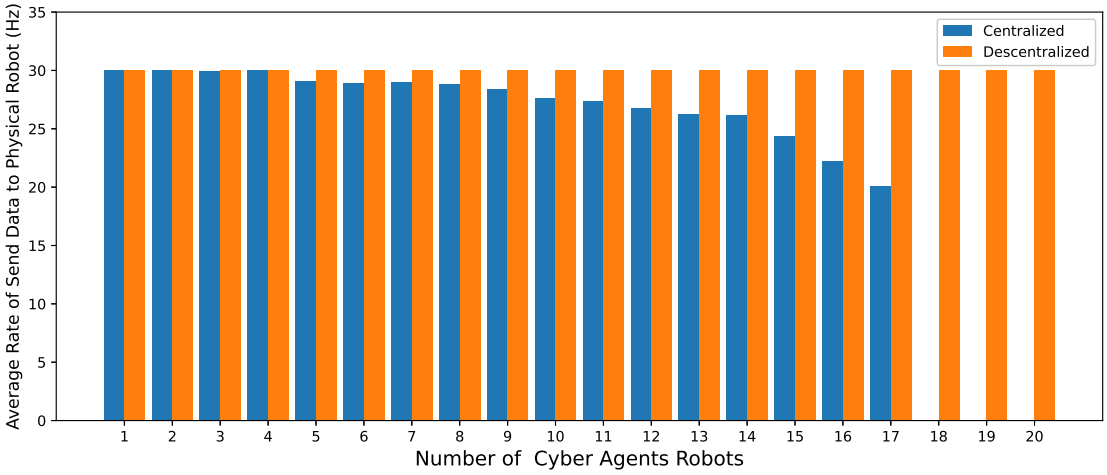
To evaluate the performance of the scalability of the propose MAS based CPS architecture, i.e., how the CPS can be enlarged without problems, it was comparing a centralized and the proposed approach. This experiment analyze the rate of sending information regarding the linear and angular velocity of the cyber layer to its respective physical layer of the agent robot. During the development of the ARENA platform, the robot's cyber layer was designed to perform its functions to send speed information to control the physical robot at a rate of 30 hertz. In other words, the cyber layer calculates the angular and linear speed of the robot, while the physical layer follows the desired route, receiving information at least 30 times per second. Therefore, when monitoring this interaction rate between cyber and physical layer, it is possible to verify if the agents fulfill the expected behavior of all their functions, such as negotiation, route planning, communication, control, and data analysis within the expected. In case of the sending data to the physical robot is delayed, consequently these functions will be in risk.

The data collection was divided into two parts, totaling forty experiments. The first corresponds to the centralization of the cyber layer of the agent robots, starting with the control of one robot up to twenty robots, totaling, therefore, twenty experiments. To centralize, cybers agents were embedded in a script executed on a single Raspberry pi 3B +. In the second part, the cyber layers of the agents were distributed, where each script responsible for an agent robot was executed in a unique Raspberry pi B +. One to twenty Raspberries were used for this second part.

To quantify the data sending rate, for each agent 2000 samples of data from the cyber layer to the physical layer of the robot were time measured. As the number of agents in the experiments increases, the 2000 samples for each agent are maintained and an average is calculated considering the increase in agents. Figure 87 presents the data resulting from the experiment.

During all tests, the decentralized proposed approach were able to maintain the desired rate of sending data to the robot's physical layer, around 30 Hz as intended. However, considering a centralized approach, there is a slight attenuation of the rate of sending data from executing five agents. This small attenuation gradually increases up to an operation of fourteen robots

Figure 87 – Response rate of the centralized vs decentralized system by increasing the number of agent robots.



simultaneously, with a frequency of $26Hz$. With fifteen robots in operation, a notable attenuation of the communication rate between the cyber and physical layer is observed. This is due to the saturation of the Raspberry pi processors, being unable to maintain the desired rate due to the high number of robots in the system. This situation occurs until the activation of seventeen robots, when the communication rate is at $20Hz$. As of the eighteenth robot, the script that centralizes all cyber agents robots, after being initialized, interrupts the microcomputer operation, which justifies the occurrence of no communication in the range of samples considered.

Therefore, the scalability system requirement based on the proposal of this thesis demonstrates satisfactory results compared to traditional centralized systems. Although scalability has been evaluated for robots, it represents the agent type that more processing can be extended to the rack/stand agents and mainly product agents that require scalability due to the flow of products within a warehouse.

D.7.2 Transporting Time Analysis

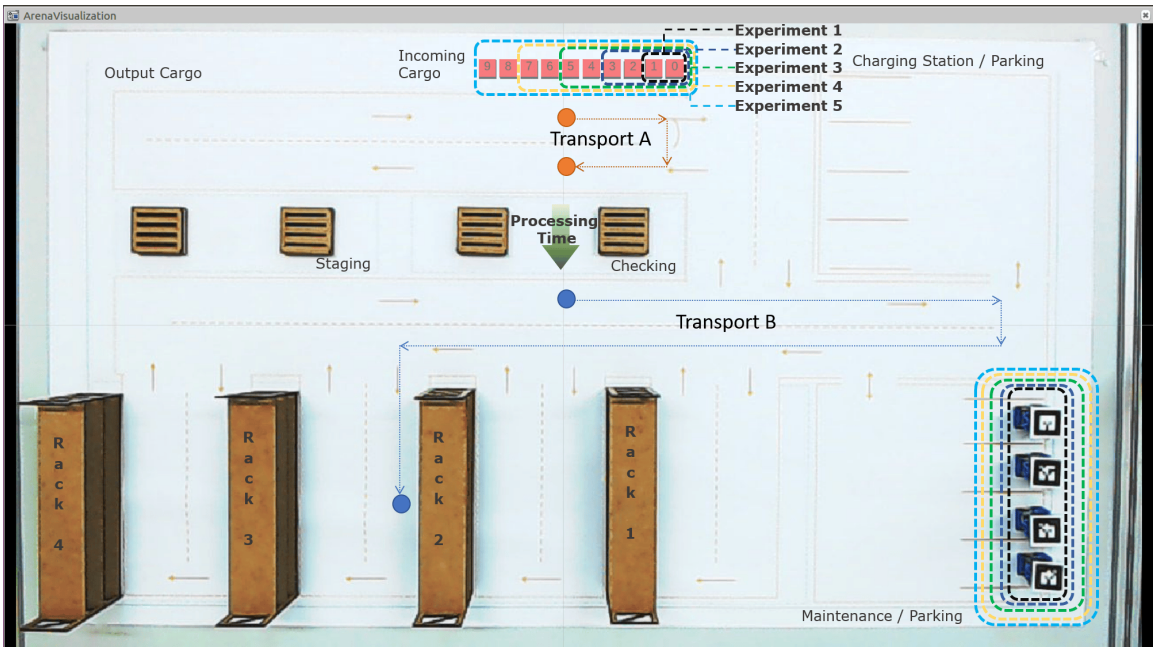
A setup was established to validate the proposed approach, checking the transport times when varying the number of products or robots. The validation consists of two stages, where the objective is to transport products from the “incoming cargo” stand to the “checking” stand (Transport A) and then to a rack where the product will be stored (Transport B).

In order to standardize, before starting each experiment, all robots receive a full battery charge to reproduce the same criteria in the negotiation between product and robot. In addition, the route to the storage from “checking” to the rack location is always the same. A adaptation

was made in the script responsible for the agent " rack 2 ", setting the bid in the auction to 0, that is, the minimum value. In this way, "rack 2" will always win the auction, maintaining the standard of distance traveled by the robot to leave the products. Finally, the processing time per product at the "checking" stand was set to 10 seconds, starting when the product arrives at this stand.

In the first stage, four agent robots were used to transport agent products that arrived at the warehouses to "rack 2", as shown in Figure 88. In this stage, five experiments were carried out, starting with two-agent products (P0 and P1), and with each new experiment, two-agent products were increased to a total of ten products (P0 to P9). Figure 89 shows the times consumed to carry the transport from the incoming cargo to the rack in detail by experiment, where "Waiting Transport" variable counts the time necessary for the product to negotiate plus the wait for the transport to be carried out.

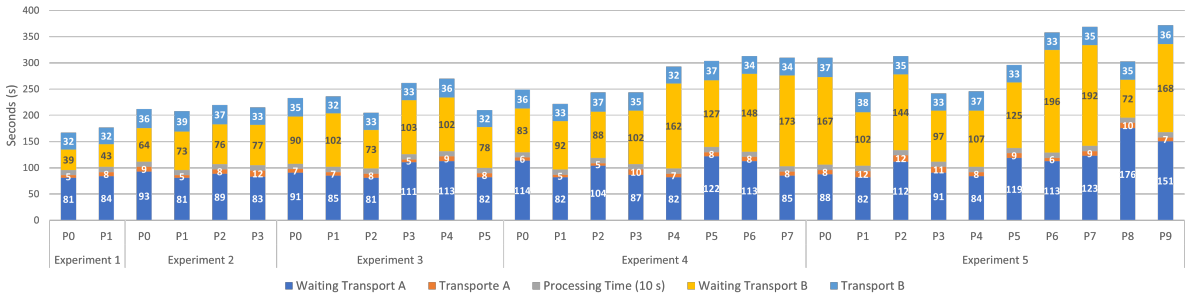
Figure 88 – Experiments setup to analyze the behavior of the MAS based CPS when increasing products agents.



It is possible to observe that Waiting Transport A depends on the number of robots per product. For experiments with more products than robots, this time tends to be longer for products that exceed the number of robots. As the number of products is greater than the number of robots, the system adapts to distribute this discrepancy in Waiting Transport A and Waiting Transport B. A special case in experiment 1 was observed, Waiting Transport B is significantly smaller than the others. Through the negotiation between product and robot, the initially idle robots were chosen to execute Transport B, optimize and justify this values. Transport A and

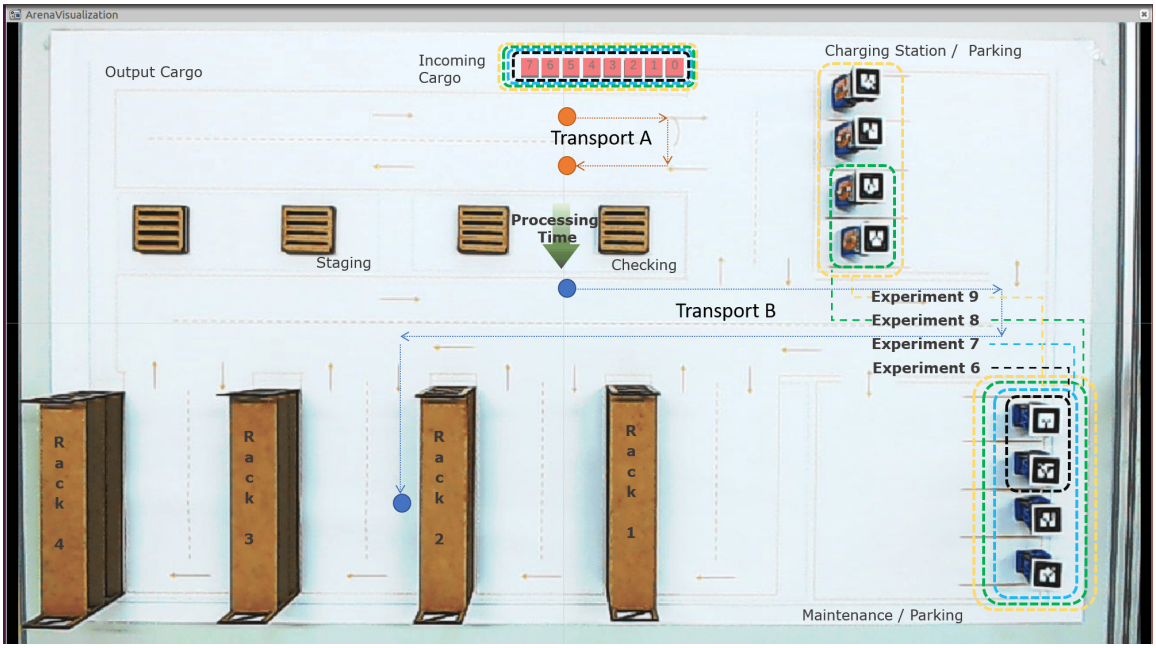
Transport B times remain similar for all cases. Therefore, the system presents robustness for the transport and the execution of the transport.

Figure 89 – Transporting time when increasing products agents.



For the second stage, a quantity of eight product agents was used (P0 to P7), which will be transported from “incoming cargo” to “rack 2”, as can be seen in Figure 90. In this stage, a total of four experiments were conducted, starting with two robot agents, with an increase of two robot agents per experiment up to a total of eight robots to carry out all eight agents products to the “rack 2”.

Figure 90 – Experiments setup to analyze the behavior of the system when increasing robot agents.



In experiment 6, the proportion of products for robots was very high, and initially, both robots went to make Transport A. When the demand for Transport B increased, one robot went to meet it while the other finished Trasport A. After all the products had been done the Transport A, the two robots were to carry out Transport B. Experiment 8 with six active robots equalized the waiting time for all products. The same situation was noted in experiment 8. It is noted that

although experiment 7 has two robots less than experiment 8 and, the distributed system was able to adapt and managed to maintain similar responsiveness.

Figure 91 – Responsiveness and flexibility analysis when increasing products agents.

Finally, Table 25 summarizes all the experiments performed that quantified transport times. Experiment 6 (two robots and eight products) had the highest value in average time related to Waiting Transport A. However, the system managed to decrease this in Waiting Transport B. For all cases, varying the number of robots or products, Transport A or Transpose B times tend to maintain a pattern. The worst case of Waiting Transport B was experiment 5 (four robots and ten products). Although there are more products than robots for transport, the system balanced Waiting Transport A and Waiting Transport B, since there was a separation, where initially the four robots went to Transport A, then two robots went to Transport B, and after Transport A finished the remaining two robots went to Transport B.

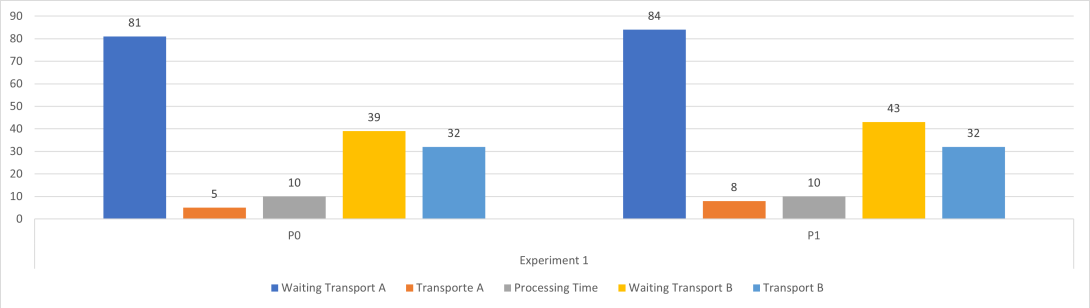
Table 25 – Statistical results of the total of 9 experiments carried out for transport time (R indicate robots and P indicate product).

	Waiting Transport A [s]				Transport A [s]				Waiting Tranport B [s]				Transport B [s]			
Experiment	avg	std	min	max	avg	std	min	max	avg	std	min	max	avg	std	min	max
1 (4R - 2P)	82.50	2.12	81.00	84.00	6.50	2.12	5.00	8.00	41.00	2.83	39.00	43.00	32.00	0.00	32.00	32.00
2 (4R - 4P)	86.50	5.51	81.00	93.00	8.50	2.89	5.00	12.00	72.50	5.92	64.00	77.00	36.25	2.50	33.00	39.00
3 (4R - 6P)	93.83	14.51	81.00	113.00	7.33	1.37	5.00	9.00	91.33	13.26	73.00	103.00	33.50	1.64	32.00	36.00
4 (4R - 8P)	98.63	16.44	82.00	122.00	7.13	1.73	5.00	10.00	121.88	35.61	83.00	173.00	34.75	1.83	32.00	37.00
5 (4R - 10P)	113.90	30.65	82.00	176.00	9.20	2.04	6.00	12.00	137.00	42.83	72.00	196.00	35.20	1.81	33.00	38.00
6 (2R - 8P)	205.25	153.00	69.00	398.00	8.38	1.92	6.00	11.00	115.63	37.85	72.00	181.00	34.13	1.81	31.00	37.00
7 (4R - 8P)	98.38	14.94	83.00	120.00	7.50	1.07	6.00	9.00	120.88	33.81	83.00	168.00	35.75	1.04	34.00	37.00
8 (6R - 8P)	98.00	12.77	79.00	118.00	10.25	1.16	9.00	12.00	89.63	12.86	78.00	111.00	37.75	2.82	35.00	42.00
9 (8R - 8P)	100.38	11.15	84.00	115.00	9.75	3.15	5.00	15.00	83.75	9.11	65.00	94.00	39.88	3.31	34.00	45.00

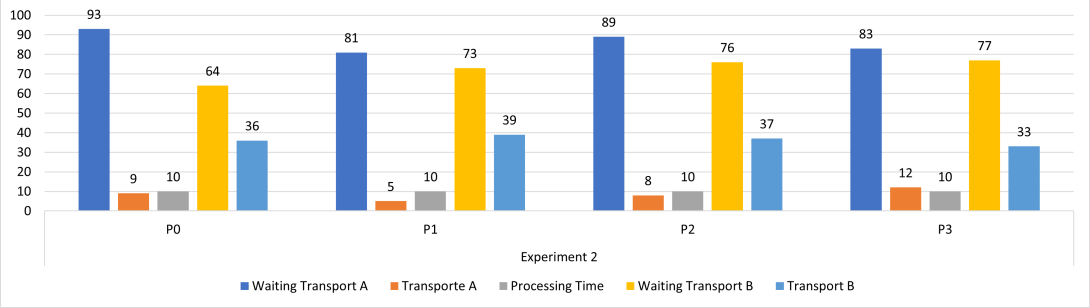
D.7.3 Transport Time Experiment varying the number of product agents

Experiment to analyze the waiting and transport times of a product, keeping the number of robots agents fixed and increasing the number of products agents.

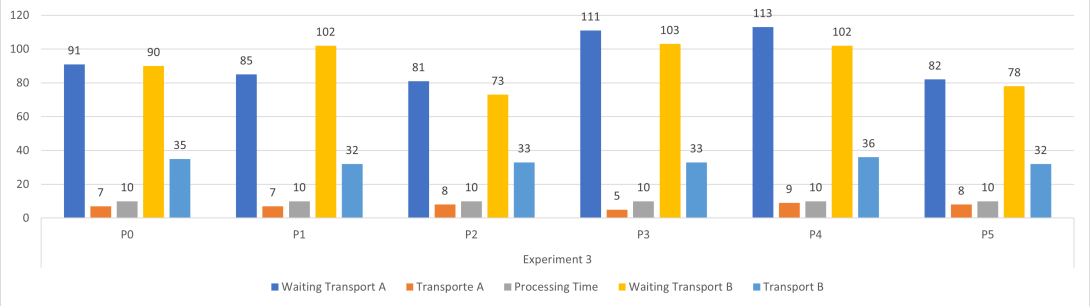
Experiment 1 with 4 Robots and 2 products.



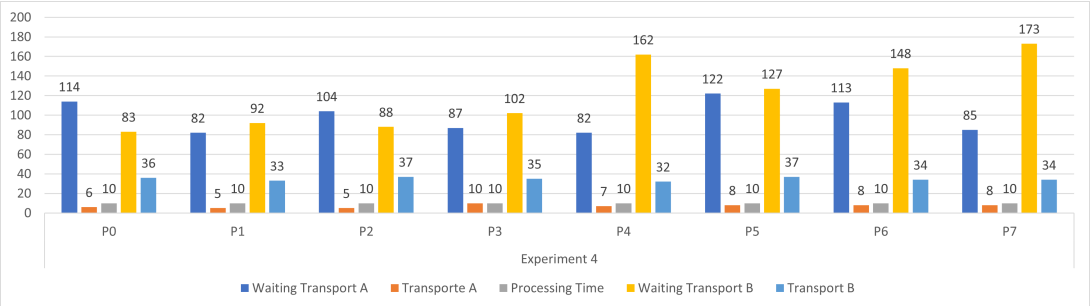
Experiment 2 with 4 Robots and 4 products.



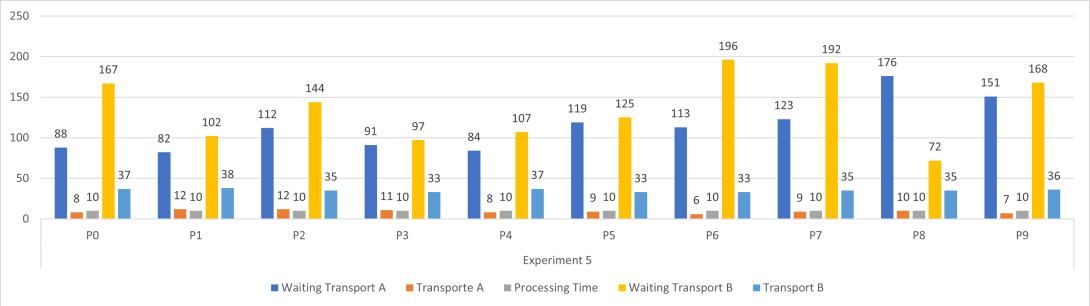
Experiment 3 with 4 Robots and 6 products.



Experiment 4 with 4 Robots and 8 products.



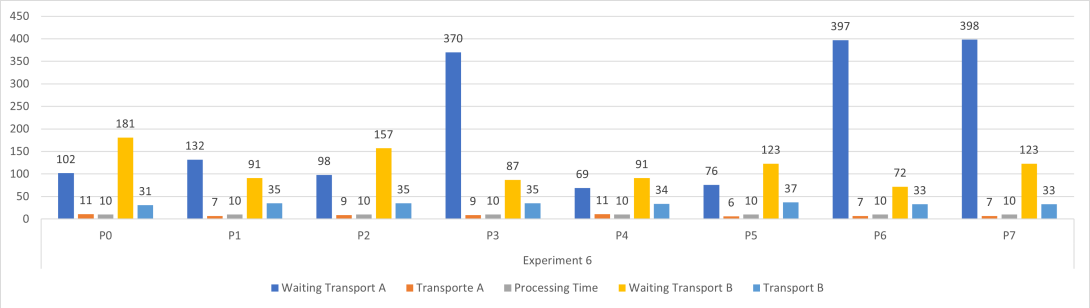
Experiment 5 with 4 Robots and 10 products.



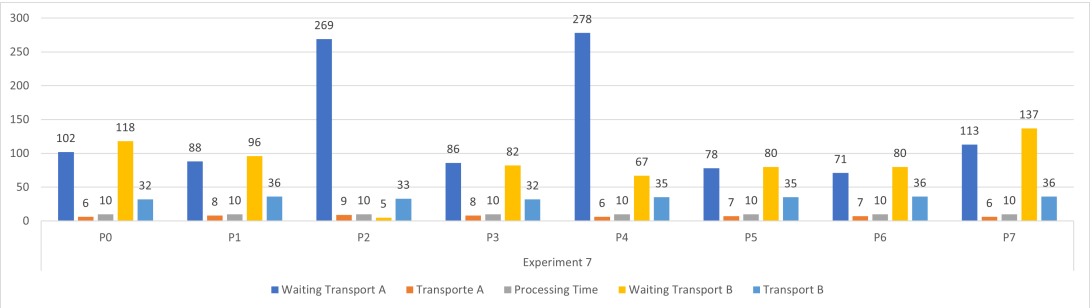
D.7.4 Transport Time Experiment varing the number of robots agents

Experiment to analyze the waiting and transport times of a product, keeping the number of products agents fixed and increasing the number of robot agents.

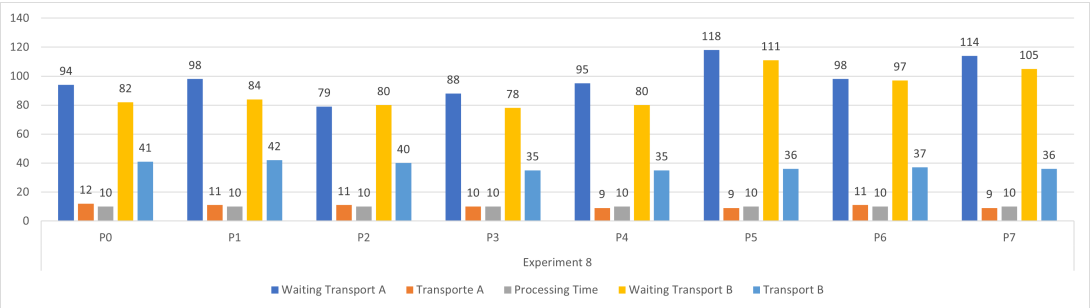
Experiment 6 with 2 Robots and 8 products.



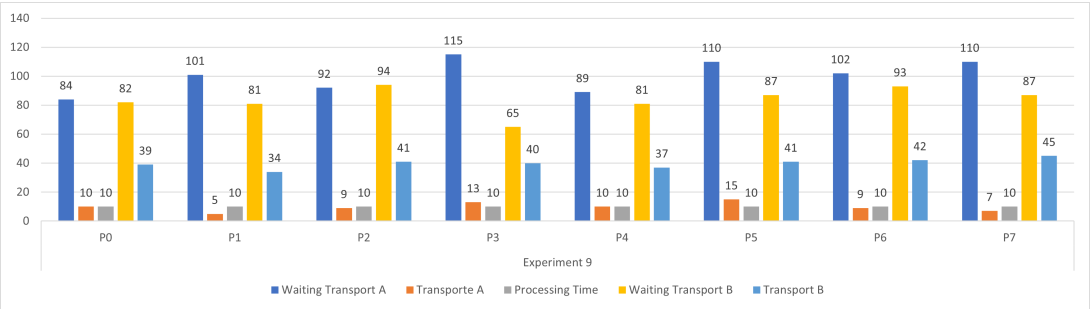
Experiment 7 with 4 Robots and 8 products .



Experiment 8 with 6 Robots and 8 products .



Experiment 9 with 8 Robots and 8 products.



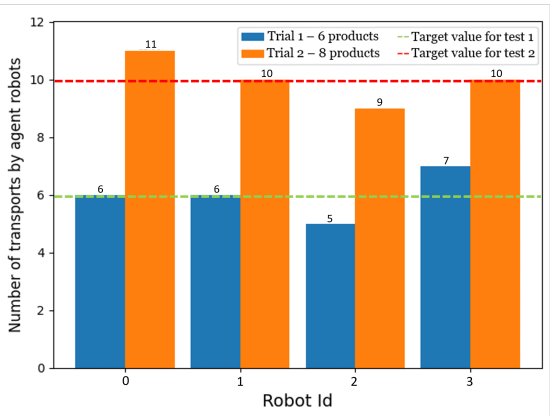
D.7.5 Resource Usage or Divisibility

Considering the existing ecosystem in the warehouse and the ability between the different types of agents to communicate, distributed coordination tends to balance flexibility, robustness, and optimization. In this way, through a market-based negotiation, the product will determine a robot to perform the transport, considering the availability, amount of battery, distance, the number of transport performed by each robot.

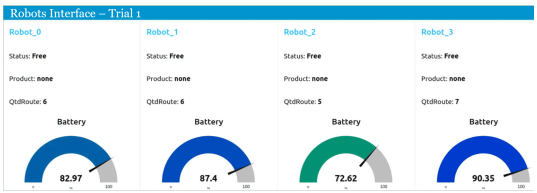
To analyze the divisibility and the use of resources, two tests were carried out with four robots. The first trial has six products, and the second has ten products. In both tests, all products take the following process plan in the warehouse: The products start at the “incoming cargo” and are transported to the “checking’ where they pass through a process that takes 10 seconds. Then, they wait for the transported to a rack, where they spend through a storage time, randomly varying from 5 to 10 seconds. Once the storage period is complete, the process of leaving the warehouses begins, requesting transport to the “staging” stand where they will be processed for 10 seconds. After processing time is over, they are transported to the “outgoing”, leaving the warehouse.

In the first trial, there are 24 transport actions (4 for each product) performed by the four robots. The second trial, there are 40 transport actions performed by the same robots. The video available in https://youtu.be/C6BdgI0akFA shows the execution of these experiments. Figure 92.a shows the number of products transported by each robot.

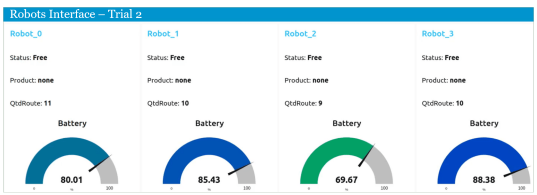
Figure 92 – Divisibility of the robot agents to carry out the transports in the warehouse.



(a) Divisibility experiment overview



(b) Arena robot dashboard in trial 1 with 6 products.



(c) Arena robot dashboard in trial 2 with 8 products

The distributed system, through negotiation governed by interaction presented in Figure 86, seek to balance the amount of transport carried out by robots, avoiding idle robots in the system, in addition to long-term wear and tear between all robots, facilitating maintenance

planning. In Figure 92.b, the robot with the least battery (robot 2) was spared from carrying out more transport, leaving it to the robot with the highest percentage of battery. In trial 2 (Figure 92.c), the robot 2 with less battery was saved again. Then, dynamically, the robot that was closest to carrying out the last action (robot 0) was compensated with the execution of the transport.

D.8 QUALITATIVE DISCUSSION

In a real logistics warehouse, the combination of complexity and dynamism represents a current challenge for implementing autonomous devices in these environments. Through the definition and classification of the main components of a logistic warehouse, enabling to understand and define the function of each component inserted in this scenario. With this in mind, and after carrying out analyzes and experiments, it is possible to improve the operation of each component of the system. The standardization of communication models is also an essential factor for interoperability between different devices of the system is essential to implement a smart warehouse.

The construction of a CPS environment with MAS cooperates to achieve logistic warehouse requirements. A balance between robustness, flexibility, responsiveness, and scalability is achieved. Scaling a distributed system becomes an intuitive process, as it is enough to replicate the desired cyber-physical component with respective agent. Through communication, other devices already operating on the system will identify the agent introduced without stopping or reprogramming the system. In addition, due to the intrinsic modularity in each agent, adding different data analysis algorithms or changing parameters in the cyber layer of each agent becomes a transparent task, as it is enough to change the code of an agent, and if it is necessary to replicate for the entire type of agents.

An unscheduled change, common in logistics warehouses, can compromise an autonomous system. However, through market-based trading, agents can regularly behave when facing a disturbance or change.Once the infrastructure for the MAS is established and the agent interaction protocol implemented, it becomes a simple task to add, adapt, or edit agent modules. The system becomes reusable and can be applied in other CPS environments, not limited to just logistical systems.

APPENDIX E – TRAINING DATASET FOR DIAGNOSIS EXPERIMENT

Figure 93 – Dataset used for $agent_1$ to training diagnosis models.

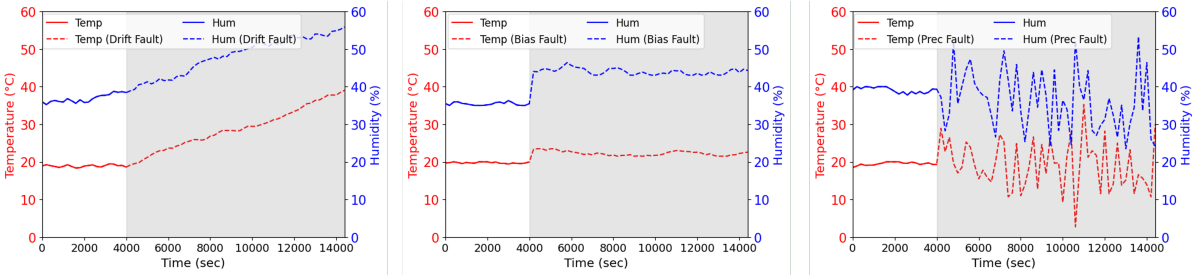


Figure 94 – Dataset used for $agent_2$ to training diagnosis models.

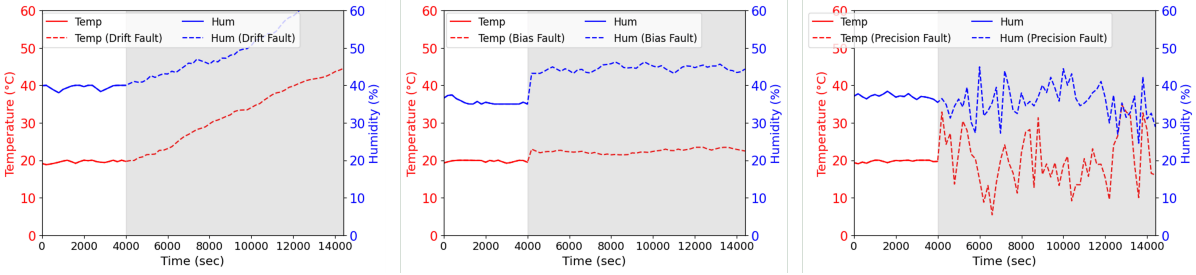


Figure 95 – Dataset used for $agent_3$ to training diagnosis models.

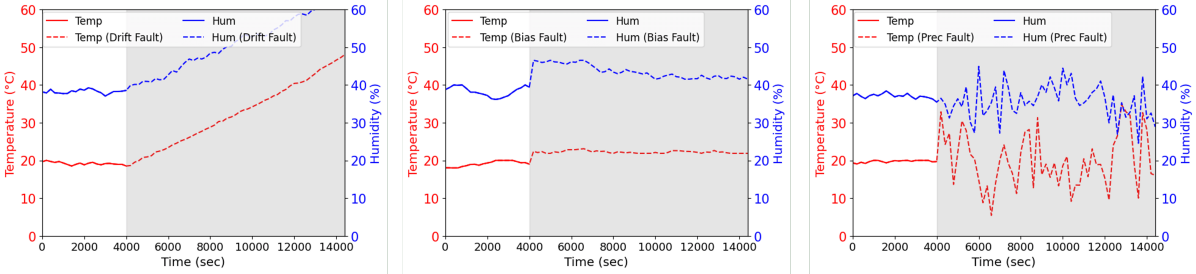


Figure 96 – Dataset used for $agent_4$ to training diagnosis models.

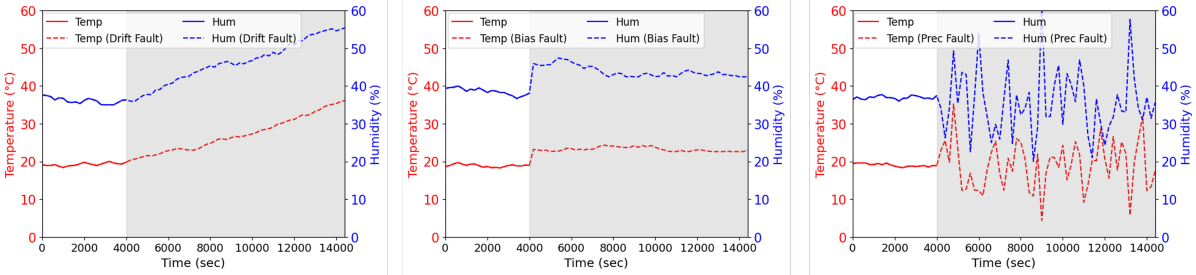


Figure 97 – Dataset used for $agent_5$ to training diagnosis models.

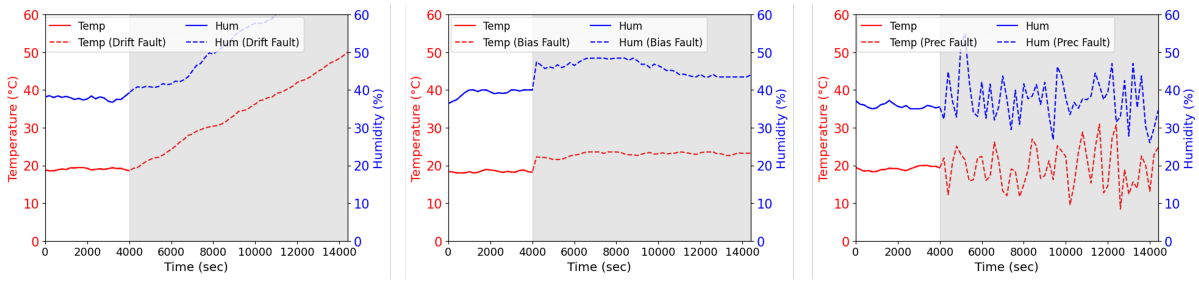


Figure 98 – Dataset used for $agent_6$ to training diagnosis models.

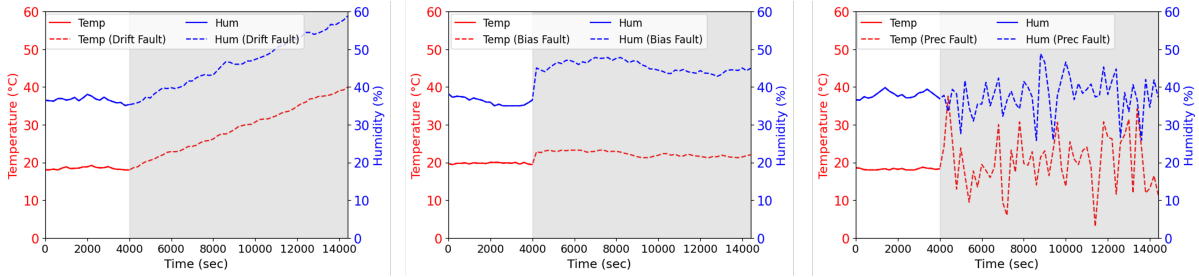


Figure 99 – Dataset used for $agent_7$ to training diagnosis models.

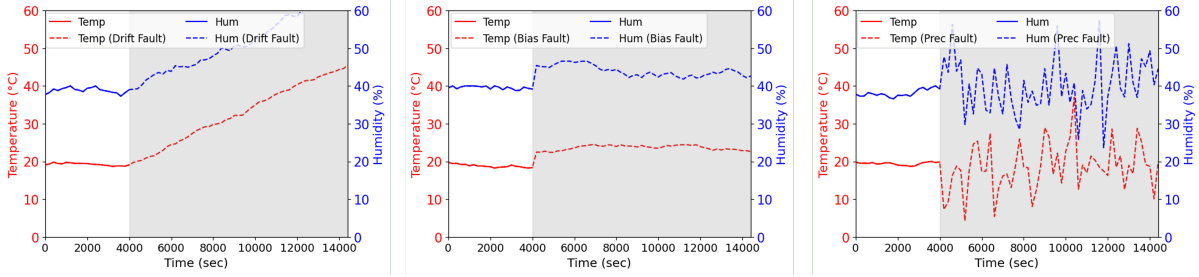


Figure 100 – Dataset used for $agent_8$ to training diagnosis models.

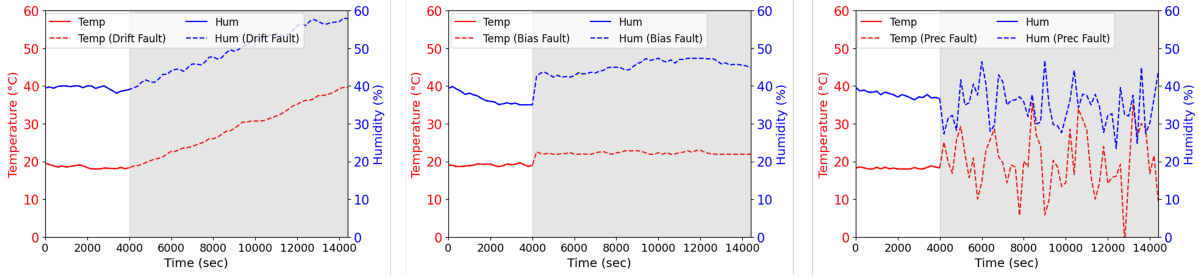


Figure 101 – Dataset used for $agent_9$ to training diagnosis models.

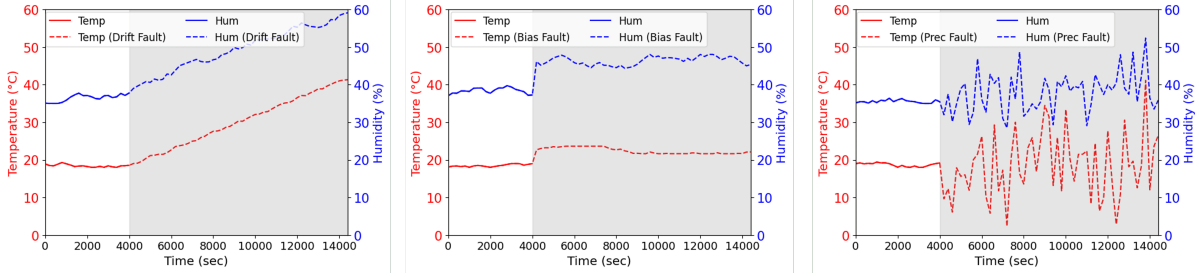


Figure 102 – Dataset used for $agent_{10}$ to training diagnosis models.

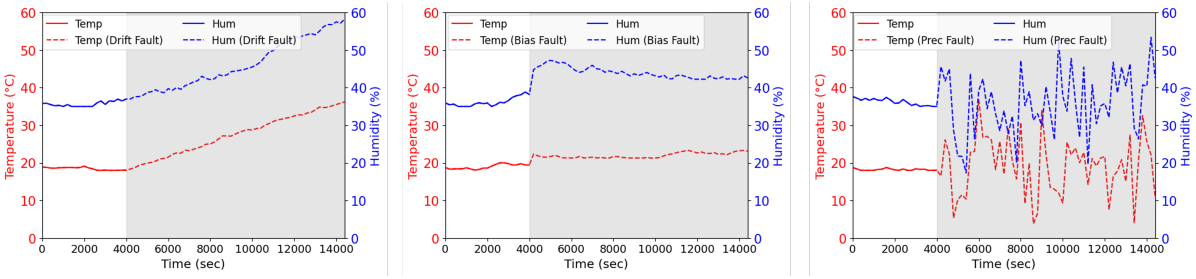


Figure 103 – Dataset used for $agent_{11}$ to training diagnosis models.

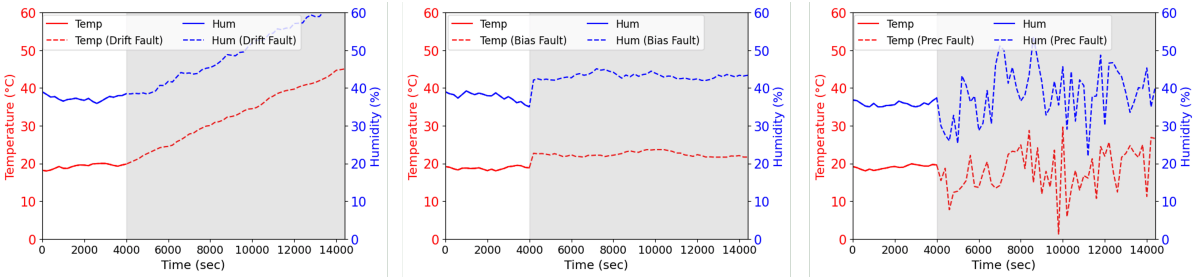


Figure 104 – Dataset used for $agent_{12}$ to training diagnosis models.