MONDRAGON UNIBERTSITATEA

GOI ESKOLA POLITEKNIKOA
FACULTY OF ENGINEERING

PHD THESIS

Facing Evolution on Industry 4.0: Modular Monitoring and Adaptive & Adaptable Visualization for Industrial Cyber-Physical Systems

Author:
AITZIBER IGLESIAS GOIKOETXEA

Supervisors:
Dr. GOIURIA SAGARDUI MENDIETA
Dr. CRISTÓBAL ARELLANO BARTOLOMÉ

Computer and Electronics Department
Faculty of Engineering
Mondragon Unibertsitatea

Arrasate
July 2019

Facing Evolution on Industry 4.0: Modular Monitoring and Adaptive & Adaptable Visualization for Industrial Cyber-Physical Systems

PhD Thesis

Mondragon Unibertsitatea

to obtain PhD Thesis in the following program

Mechanics Engineering and Electrical Energy PhD Program

Author:

Aitziber Iglesias Goikoetxea

Supervisors:

Goiuria Sagardui Mendieta
Cristóbal Arellano Bartolomé

Arrasate, July 2019

Facing Evolution on Industry 4.0: Modular Monitoring and Adaptive & Adaptable Visualization for Industrial Cyber-Physical Systems

"Perché la minestra
si fredda ...ecc,ecc. ..."

LEORNARDO DA VINCI

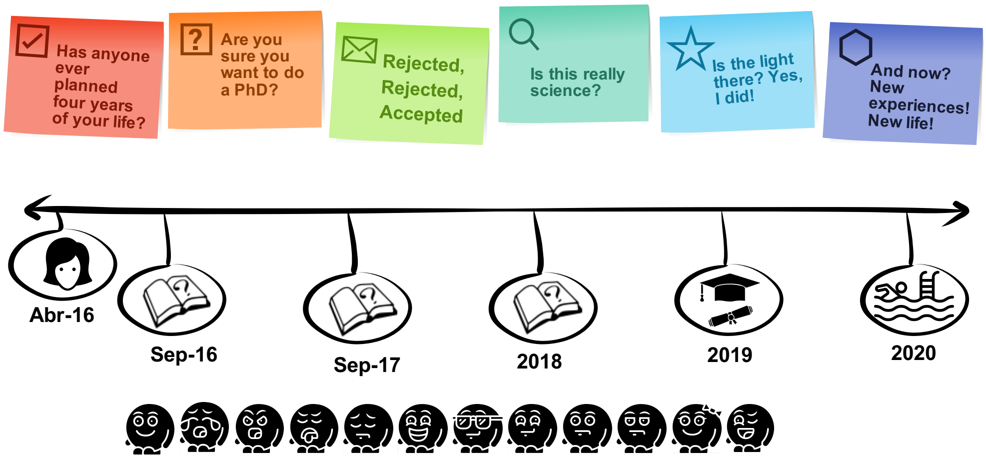
Nire guraso eta mutilari
Aitziber

Acknowledgements

I have waited until the last moment to write this part of the dissertation... the moment to start remembering all the things that I have been living during the last three years. A white sheet of paper, without words and... what can I write? Who can I thank and how? A big dilemma... My stage began on April 4, 2016, where Salva put four years of my life on a whiteboard and asked me... had anyone ever planned 4 years of your life? My answer was evident... NO! But, something happened at the time that I said... why not doing a Ph.D.? Why not?

I started at Ikerlan as a researcher and people asked me, are you sure you want to do a Ph.D.? I don't really think I was sure about it, but... who never takes risks never wins, hence, after some inquiries... I jumped into this research world. They say that reality surpasses fiction, that's how it has been, reality was different, I didn't know where I was getting myself into. A period where I have known myself better and although there have been many "bad" moments I wouldn't change it. Those rejects one after the other, the ridiculous questions trying to figure out if what I was doing was valuable or not. Think about science, what is science? But in the end ... I saw the LED that was lighting up very slowly and by the time I realized... that years passed, and another part of my life just passed. And I ask myself, what now? Well, keep on living experiences, keep on meeting people and start a new stage, but without forgetting the past. That is why know I need to look back and thank everyone who has been important to me.

First, I would like to start with my supervisors, Goiuria and Cristobal. What to say... My Ph.D. wouldn't be the same without turning back and forth the document. Or those intense and desperate conversations with Cristobal. I cannot forget those who have taught me so much, Tao and Shaukat. You have brought out a different side of me, you have made me believe in myself and trust the work I was doing. I would like to continue with the PEOPLE from IKERLAN, all those who have walked



with me and endured throughout the thesis. My colleagues, Josu, Aizea, Leti, Xabi X2, Nerea, Mikel, and the biggest one Ángel, I am still waiting for the "batamanta". I cannot forget either the rest of the people from "database architectures" or "IoT cyber secure" that of secure... hahaha. Among them, I need to highlight Karmele and Salva. Someone perhaps is getting a bit nervous reading these words, but not, I haven't forgotten about you, my runner boys Marc, Iker and Aitor. Mordor people, Lorea and Joseba. Those who have taken me home a thousand times, Marco and Victor (Pepa & Abelino). The blonde of the team, Idoia, and especially to Joserra. Eskerrik asko (thank you)! I also need to thank the person who left his mark after his stay in IKERLAN, a great person Luis. All of you have made it possible to deposit the thesis and bring this stage to an end. Mila esker!

My KUADRILLA, What would I do without my "PENDEJAS"?, eskerrik asko bihotz bihotzez, even if I disappear many times from the map, you have always been there to support and encourage me. How to forget the hangouts of Amaia Casta or the audios of Beni and Dei. Or all the encouragement messages from Korta, Sara, Merin, and Larre. Those days where you want to clear yourself, go out to the street relax with Naia and Maite or having alternative conversations with Helen, Eñaut, and Fast. I cannot forget anybody from "Pipar bete jana" who have always been there. Benetan Eskerrik asko!

I cannot forget those people who accompanied me on my stay, especially Mary Jane, a great treasure hidden in the cold lands of Norway. A person of 10, a sister, a lifelong friend.

Aita, Ama, what to say? I do not have words to thank you for everything, thank you for putting up with my bad days, my anger, my bad faces, my bad answers. But *without you, I would have never reached where I am. Eskerrik asko for supporting me in everything. My Gramma, a special person in my life, an inspiring example to follow and my greatest reference. A grandmother who doesn't give up and always fights until the end. And others who are no longer here but I will never forget, Uncle Dioni and Uncle Venancio.*

And finally... that special person who has appeared in my life, el sonrisas, my travel partner, of tears and laughter. Eskerrik asko Raül for all your support, for all you have taught me and for being able to make me explode when I need it most and then bring out a big smile. Moltes Gràcies for conquering me and all around me. You're the best!

To all of you ...
*... **THANK YOU** ...*
... Because without you ...
... none of this would make sense!

Declaration

Hereby I declare that this document is my original authorial work, which I have worked out on my own. All sources, references, and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

Aitziber Iglesias Goikoetxea

Oñati, July 2019

This work has received funding from the Electronic Component Systems for European Leadership Joint Undertaking under the MegaM@Rt2 project (Grant agreement No. 737494) in the EU Horizon 2020 program and the Basque Government through the Elkartek program under the TEKINTZE project (Grant agreement No. KK-2018/00104).

Laburpena

Industry 4.0 sona handia hartzen ari da egungo gizartean. Hori dela eta, zeinbat esparru industrialetan, sistema ziberfisiko industrialen (ingelesez, Industrial Cyber-Physical System, laburdura ICPS) monitorizioak berebiziko garrantzia hartu du. Guzti honen funtsa, datuen bilketa era eraginkorrean egitean datza, ondoren datu hauek analizatu eta sistema industrialetan eragin dezaketen erabakiak hartzeko asmoz. Industrial Cyber-Physical Systems (ICPSs)-ak eboluziona dezaketen gailu heterogeneo, banatu eta autonomoez osatuta daude eta, ondorioz, monitorizazio sistemak egokitzea ezinbesteko bihurtzen da gaur egun.

Tesi honek, eboluziona dezaketen ICPSs monitorizazio sistema bat aurkezten du, zeina hainbat domeinu industrialetarako diseinatu den. Proposamen honetan, datuen bilketa eta biltegiratze eraginkorra eta eboluzioen hautematea eta informazioaren bistaratzea aztertzen dira. Proposamena bi azpisistemaz osatuta dago: (I) *Modular Monitoring System*, hainbat estandar bateratzetik sortu den sistema bat da, non, datuak jaso eta berauek era estrukturatu batetan gordetzeko gaitasuna duen; eta (II) *Personal Visualization & Evolution Detection System*, erabiltzaileak bere interfazeak sortzeko gaitasuna duen sistema bat da eta, horrekin batera, eboluzioak detektatu eta hauen inguruan alertak sortzeko gaitasuna du. Proposamena balioztatzeko asmoz, sistema bakoitzaren prototipo bat sortu eta ebaluatu da tesi honetan.

Abstract

Industry 4.0 comes to play an important role in various industrial domains where monitoring industrial cyberphysical systems (ICPSs) is becoming essential. This is due to the necessity to efficiently collect data from industrial processes for then making decisions that can impact on the operation of the industrial systems. Typically, the ICPSs are composed by heterogeneous, distributed and autonomous physical devices which can evolve over time. Thus, this makes necessary the adaptation of the monitoring system according to the physical devices.

In this dissertation, it is proposed a monitoring system for ICPSs that evolve over time, that has been designed for multiple domains. In this proposal, the data capture and storage beside the ICPS evolution detection and information visualization are considered. To do so, the proposed solution is composed by two subsystems: (I) *Modular Monitoring System* which is based on the union of different standards able to capture data and store it in a structured manner and; (II) *Personal Visualization & Evolution Detection System* where the user has the possibility of customizing its visualization and the system is able to trigger alerts on ICPS evolution. In order to validate the proposal a prototype of each system has been developed and finally evaluated.

Resumen

La Industria 4.0 juega un papel importante en diversos ámbitos industriales donde la monitorización de los sistema ciber-físicos industriales (en inglés, Industrial Cyber-Physical Systems, abreviadamente ICPSs) se está convirtiendo en parte esencial. Esto se debe a la necesidad de recolectar datos de los procesos industriales de manera eficiente para luego tomar decisiones que puedan impactar en el funcionamiento de los sistemas industriales. Normalmente, los ICPSs están compuestos por dispositivos físicos heterogéneos, distribuidos y autónomos que pueden evolucionar con el tiempo. Por lo tanto, esto hace que sea necesaria la adaptación de los sistemas de monitorización de acuerdo con los dispositivos físicos.

En esta tesis se propone un sistema de monitorización para ICPSs que evolucionan con el tiempo, el cuál ha sido diseñado para múltiples dominios. En esta propuesta se considera la captura y almacenamiento de datos además de la detección de la evolución y la visualización de la información. Para ello, la solución propuesta se compone de dos subsistemas: (I) *Modular Monitoring System* que se basa en la unión de diferentes estándares capaces de capturar datos y almacenarlos de forma estructurada y; (II) *Personal Visualization & Evolution Detection System* donde el usuario tiene la posibilidad de personalizar su visualización y el sistema es capaz de generar alertas en caso de que se produzca una evolución. Para validar la propuesta se ha desarrollado un prototipo de cada sistema y finalmente se ha evaluado.

Contents

I Introduction **1**

1 Overview . . . **3**

1.1 Motivation and Scope of the Research . . . 4

1.2 Research Methodology . . . 5

1.3 Glossary . . . 7

1.4 Research Hypothesis . . . 8

1.5 Contributions . . . 8

1.6 Publications . . . 9

1.7 Research Activities . . . 10

1.7.1 Research Support . . . 10

1.7.2 Research Stay . . . 12

1.8 Outline . . . 12

II Foundation & Context **15**

2 Background . . . **17**

2.1 Industry 4.0 & Cyber-Physical Systems . . . 18

2.2 Monitoring Evolving Industrial Cyber-Physical Systems . . . 23

2.3 Managing ICPS Evolution: Comparison Approaches . . . 24

2.4 Adaptive & Adaptable User Interfaces for Industry . . . 25

3 Problem Statement . . . **29**

3.1 Introduction . . . 30

3.2 Domain Analysis . . . 32

3.2.1 Industrial Domains Information . . . 32

3.2.2 Domain Analysis Outcomes . . . 42

3.3 Revisiting Requirements through a Systematic Literature Review . . . 47

3.3.1 Process for a Systematic Evaluation of the Primary Studies . . . 48

3.3.2 Results of the Systematic Literature Review . . . 52

III Contribution **59**

4 Modular Monitoring System . . . **61**

4.1 Introduction . . . 62

4.2 Modular Monitoring System 66
4.3 Modular Monitoring System Components 68
4.3.1 ICPS Composition based on ISA-95 68
4.3.2 Agent-Based Communication 69
4.3.3 Data Representation by IEC 61850 and IEC 62264 72
4.4 Modular Monitoring System Configuration 75
4.4.1 Agent Generation 75
4.4.2 Monitoring System Configuration 77
4.5 Conclusion 80
5 Personalized Visualization & Evolution Detection System 83
5.1 Introduction 84
5.2 Personal Visualization & Evolution Detection System Components . 85
5.3 Creating GEs at Runtime by the End User 90
5.4 Automatic Detection and Classification of ICPS Evolution 91
5.4.1 Data Model Comparator 92
5.4.2 Evolution Classification 94
5.5 Adapting Dashboards to users 96
5.6 Conclusion 99
IV Development & Evaluation 101
6 Prototype & Evaluation 103
6.1 Prototype & Validation 104
6.1.1 Modular Monitoring System 105
6.1.2 Semiautomatic Agent generation 108
6.1.3 Personalized Visualization & Evolution Detection System . 112
6.2 Evaluation 116
6.2.1 Return Of Investment 116
6.2.2 Survey about proposed solution 120
6.2.3 Scalability test 125
6.3 Applying Solution into other domains 134
V Final Remarks 137
7 Conclusion 139
7.1 Summary of the Contributions 140
7.1.1 Hypothesis Validation 141
7.1.2 Limitations 143
7.2 Lessons Learned 143
7.3 Perspectives and Future Work 146
7.3.1 Short-term Perspectives 146

7.3.2 Medium-term Perspectives 147
Bibliographic References 149
VI Appendix 163
Appendix A Industrial Domain Workflows 164
A.1 Automated Warehouse Workflow 164
A.2 Press Machine Workflow 165
Appendix B International Electrotechnical Commission (IEC) 61850 standards 168
Appendix C Questionnaire-based survey 171
C.1 Industrial Domain Information 171
C.2 Modular Monitoring System 183
Appendix D Scalability Test Results 185
D.1 Data Model Comparator Results 185
D.2 Data Model Sizes 188

List of Figures

1.1 Dissertation methodology . . . 6
1.2 IKERLAN Technology Research Centre . . . 10
1.3 Mondragon Goi Eskola Politeknikoa . . . 10
1.4 Simula Research Laboratory . . . 10
1.5 Dissertation outline schema . . . 13

2.1 Four stages of the Industrial Revolution. . . . 18
2.2 ISA-95 standard, four automation levels . . . 21
2.3 IEC 62264 standard: Left) The information model. Right) Information model example [IEC13b] . . . 22

3.1 Domain Analysis Process . . . 31
3.2 Example of an Automated Warehouse layout (Partial) . . . 33
3.3 Catenary-Free Tram partial Layout . . . 36
3.4 Press Machine composition: Left) Main components. Right) Specific layout example . . . 39
3.5 Similar data Structures across the domains . . . 43
3.6 The SLR process . . . 50
3.7 Left) excluded papers in technical contribution analysis (Step 4). Right) excluded papers in authors and reference analysis (Step 5) . . . 53

4.1 All data stages from its capture to its visualization in order to supervise ICPSs. . . . 62
4.2 Solution Overview . . . 64
4.3 Conceptual Model of the *Modular Monitoring System* . . . 67
4.4 Agent-Based Communication between ICPS and monitoring software system . . . 69
4.5 Agent Composition . . . 70
4.6 TRILATERAL components . . . 72
4.7 Data Model Structure based on the IEC 61850 and IEC 62264 Standards 73

4.8 TRILATERAL workflow 76
4.9 Server definition feature model 76
4.10 Modular Monitoring System Configuration and TRILATERAL 78
4.11 Sequence diagram of how a software monitoring system is configured and how it works 79

5.1 Personalized visualization & evolution detection system overview . . . 84
5.2 Main Components of the *Personalized visualization & evolution detection system* 85
5.3 Selection MetaModel 86
5.4 GE Configuration MetaModel 87
5.5 Relation MetaModel 88
5.6 GE Model 88
5.7 Operation MetaModel 89
5.8 User Creating a New GE 90
5.9 Data Model evolution example: a) before evolution, b) after evolution . 93
5.10 Diff Model creation diagram 95
5.11 Diff MetaModel Structure for alerting users about ICPS evolution . . . 96
5.12 Managing ICPS changes at runtime 97
5.13 Type of Alerts a user can receive depending on ICPS evolution 98

6.1 Prototype architecture 104
6.2 Configured ICPS structure 106
6.3 Screenshot of sof**T**ware p**R**oduct l**I**ne based mu**L**tidomain iot **A**r**T**ifact g**E**neration for indust**R**i**AL** cps (TRILATERAL). 109
6.4 Layers of the TRILATERAL tool. 110
6.5 Different protocol sizes 112
6.6 The Dataset Structure from the Real Automated Warehouse (AW) to validate the *Personalized Visualization & Evolution Detection System* . 114
6.7 Alerts when ICPS evolves. Left) before evolution. Right) after evolution (new *LogicalDevice* is introduced) 115
6.8 Development and Commissioning time currently required for 1) a new ICPS installation or 2) the introduction of a new device into ICPS according to the technical manager in AW domain 118
6.9 Comparing the *personalized visualization & evolution detection system* with the Current Practise 120
6.10 Participants' Roles 121
6.11 Knowledge about Data Visualization and Monitoring System 122

6.12 Importance of Visualizing Data 123
6.13 Challenges displaying data at runtime 123
6.14 Detecting ICPS evolution at runtime or automatically 124
6.15 Execution time average in different ICPS Data Model sizes 127
6.16 Increase the percentage of changes in different ICPS Data Model sizes . 129
6.17 Time need for detection versus ICPS Data Model sizes (the axis are in logarithmic scale) 130
6.18 Time growth trend when increases the number of devices (the axis are in logarithmic scale) 131
6.19 The relation between different Data Model file sizes and the elapsed execution time (the axis are in logarithmic scale) 132
6.20 Diagram of activity in which it is estimated to reduce the result time of the alert module. 133

A.1 Workflow of an Automated Warehouse 165
A.2 Working Process of a Press Machine 166

B.1 The information model of the IEC 61850 standard. 169

List of Tables

2.1 Comparison between different comparison solutions 24
2.2 Comparison between Adaptive and Adaptable User Interfaces (UIs) . . . 27

3.1 Characteristics of the Automated Warehouses 34
3.2 User Roles for Automated Warehouse Supervision 35
3.3 Catenary-free Tram User Roles 37
3.4 Characteristics of the Press Machines 40
3.5 User Roles for Press Machine Supervision 41
3.6 Correspondence between RQs and Comparison criteria 49
3.7 SLR results 54

4.1 Data MetaModel level description 73
4.2 An Example Data Model 75
4.3 Achieved requirements by the Modular Monitoring System 81

5.1 Models dependent on users or ICPS evolution 89
5.2 Dependencies between models 90
5.3 JaVers output depending on the input format 94
5.4 Description of the information saved in each Diff Model 97
5.5 Achieved requirements by the *personalized visualization & evolution detection system* 100

6.1 Libraries and apps needed by TRILATERAL. 111
6.2 Max and Min execution time difference percentage 128
6.3 Difference between the minimum (remove) and maximum (insert) execution time 130

7.1 Identified Requirements VS Proposed solution 140

D.1 Obtained results from the data model comparator with **50 Logical Nodes**. Average time (AVG) and Confidence Interval (CI) of execution time needed (ms) 185
D.2 Obtained results from the data model comparator with **100 Logical Nodes**. Average time (AVG) and Confidence Interval (CI) of execution time needed (ms) 185
D.3 Obtained results from the data model comparator with **500 Logical Nodes**. Average time (AVG) and Confidence Interval (CI) of execution time needed (ms) 186
D.4 Obtained results from the data model comparator with **1000 Logical Nodes**. Average time (AVG) and Confidence Interval (CI) of execution time needed (ms) 186
D.5 Obtained results from the data model comparator with **5000 Logical Nodes**. Average time (AVG) and Confidence Interval (CI) of execution time needed (ms) 186
D.6 Obtained results from the data model comparator with **10000 Logical Nodes**. Average time (AVG) and Confidence Interval (CI) of execution time needed (ms) 187
D.7 Data Model sizes in MB for comparison 188

Acronyms

CPS Cyber-Physical System

ICPS Industrial Cyber-Physical System

AW Automated Warehouse

CFT Catenary-Free Tram

PL Press Line

IEC International Electrotechnical Commission

CNC Computer Numerical Control

SCADA Supervisory Control And Data Acquisition

PLC Programmable Logic Controller

MES Manufacturing Execution System

ERP Enterprise Resource Planning

CRM Customer Relationship Management

SCM Supply Chain Management

DSL Domain Specific Language

UI User Interface

GE Graphical Element

UML Unified Modeling Language

HCI Human-Computer Interaction

HMI Human Machine Interaction

MBE Model-Based Engineering

SPL System Product Line

DDS Data Distribution Service

SOAP Simple Object Access Protocol

CoAP Constrained Application Protocol

QBS Questionnaire-Based Survery

HTTPS Hypertext Transfer Protocol Secure

HTTP Hypertext Transfer Protocol

IoT Internet of Things

SLR Systematic Literature Review

TRILATERAL sof**T**ware p**R**oduct l**I**ne based mu**L**tidomain iot **A**r**T**ifact g**E**neration for indust**R**i**AL** cps

PART I

INTRODUCTION

"Failure is simply the opportunity to begin again, this time more intelligently."

HENRY FORD

Chapter 1

Overview

This chapter provides an overview of the research conducted by this dissertation. First, the main motivation and scope of the research followed by the research methodology is introduced. Then, a glossary with the main concepts used during this dissertation are presented. Next, the research hypothesis and the contributions are described. Then the publications are introduced followed by the research activities. Finally, the outline of the document is presented.

Contents

1.1 Motivation and Scope of the Research **4**
1.2 Research Methodology **5**
1.3 Glossary **7**
1.4 Research Hypothesis **8**
1.5 Contributions **8**
1.6 Publications **9**
1.7 Research Activities **10**
1.7.1 Research Support 10
1.7.2 Research Stay 12
1.8 Outline **12**

1.1 Motivation and Scope of the Research

Cyber-Physical Systems (CPSs) play an important role as Industry 4.0 is increasingly gaining strength [LCK16, LBK15, RJHA18]. CPSs are *"physical, biological and engineered systems whose operations are monitored, coordinated, controlled and integrated by a computing and communication core"* [RLSS10]. In this dissertation it is referred as Industrial Cyber-Physical System (ICPS) since the focus is on industrial setting. An ICPS is composed of different types of devices, i.e. actuators, displays, and sensors. Monitored devices provide data, which is gathered, to be transformed into information through an User Interface (UI) in order to supervise the ICPS by the end user [ILA+17, LKY14a, LBK15].

Monitoring enables the transition from a traditional industrial system to an ICPS in the context of Industry 4.0, hence, monitoring solutions in Industry 4.0 allow automatization in an distributed, data intensive and dynamic environment. First, they receive large amounts of data coming from heterogeneous, distributed and autonomous devices [LKY14b, LBK15]. Second, the supervision of these ICPSs is managed by different types of users [ABY14b, ABY14a] that require information in order to identify anomalies during operation, make decisions, etc [BBA+17, SDSS16]. Third, each device sends different data depending on the status of the ICPS. Four, due to ICPS maintenance, retrofitting, resource optimization or because the system needs to be adapted to new services, the ICPSs evolve [GPM14, GBH+16, LCK16]. ICPS evolution refers to changes that can occur during the operation of the ICPS, i.e., devices that send different attributes depending on the ICPS state; or insertion, modification or removal of devices in the ICPS to monitor.

Variability exists, i.e., not all ICPSs are composed by the same devices. Additionally, for a given device, its logic can be different depending on the state of the ICPS which is being supervised. This can cause an evolution on the ICPS, because the data received is different making the monitoring system to evolve in order to start capturing data that was not being received before. Furthermore, the insertion of new devices or removal or modification of existing ones also provokes evolution [GPM14, GBH+16, LCK16]. This evolution needs to be manage at runtime automatically in both data collection and visualization since managing changes manually is error-prone [TLWD18, BMC+17]. In order to supervise efficiently an ICPS is important to address the next challenges:

- **Managing data acquisition on evolving ICPSs:** Considering the quantity, the diversity of the autonomous devices and the heterogeneity of the data interchanged, the ICPS evolves during operation; it is therefore a challenge to adapt at runtime the monitoring system.

- **Detecting and classifying evolution at runtime:** Considering that an ICPS evolves and the ICPS is composed by a high quantity of devices, it is a challenge to detect and classify automatically such changes and communicate them to the end users at runtime.

- **Personalizing user interfaces:** As an ICPS can evolve, adapting the UI is necessary in order to be aware of what is happening. Thus, an adaptive and adaptable visualization system is necessary, since different user roles are supervising the ICPSs and each one can have different needs. Adaptive visualization refer to UIs that change their appearance based on a algorithm, instead, adaptable systems are systems configured by the end user. Therefore, it is a challenge to identify and manage how an UI can be adapted to (adaptive) and by (adaptable) the end user in an evolving ICPS in order to help him/her make decisions.

1.2 Research Methodology

The applied methodology used in this work has been the "*Design Science Research Process Model (DSR Cycle)*" (see Figure 1.1) [VK04]. This methodology is an adaptation of the design process model developed by Takeda, et al. in 1990 [TVY90] and is based on knowledge, i.e., "knowledge is used [creatively] to construct (create) works, and works are evaluated to build knowledge" [VK04]. The methodology is composed by five different process steps which are cyclical:

- **Awareness of Problem:** The knowledge of an interesting research problem can come from: new industry developments, identification of problems within a reference discipline or even an allied discipline that provides the opportunity to apply new findings to the researcher's field, etc. All of them aim to detect an interesting problem, hence, the output of this step is a *proposal* for the identified problem. In this dissertation, different industrial domains are analyzed to identify interesting research problems.

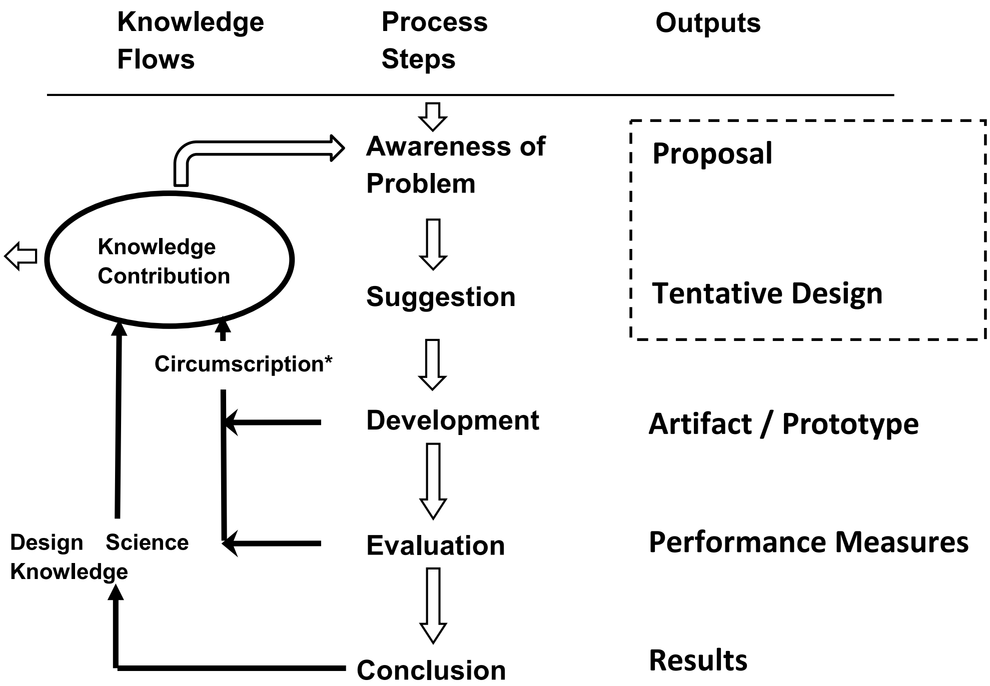


Figure 1.1: Dissertation methodology

- **Suggestion:** In this step, a novel proposal for solving the identified problem is suggested. This phase consists of taking a creative step to solve the detected problem, in which a new functionality is conceived. For it, a novel configuration of either existing, or new and existing elements is proposed. The output of this step is a *tentative design*, where the proposed design to give response to the identified problem is reflected. In this dissertation, two systems are proposed in order to give response to the identified problems.

- **Development:** The proposed design (*tentative design*) is put into practice, i.e. it is developed. The tentative design developed is known as an *artifact* or *prototype*. Notice, that the novelty is mostly in design, not in the developed *prototype*. In this dissertation, one *prototype* for each system is developed.

- **Evaluation:** In this step the *prototype*s are evaluated. To do so, different methodologies can be used depending on the prototype. The main idea is to test the presented hypothesis. The output of this phase is named as *performance measures*, where different forms can be use, such as, comparison of the artifact with other solutions, objective quantitative measures of performance, satisfaction surveys, customer feedback or simulations. In this dissertation, customer feedback, satisfaction surveys and scalabily tests are used to evaluate the *prototype*s.

- **Conclusion:** This step is the end of the research cycle. It is the phase where the tentative design, the prototype and the conducted evaluation are analyzed, i.e., it is assessed whether the design, the prototype and the evaluation are sufficient to solve the identified problem. The *results* (the output of this step) are consolidated and detailed with the aim of disseminating the knowledge obtained. In this dissertation, in order to close the cycle, the learned lessons are presented.

1.3 Glossary

In this section, different definitions are described in order to clarify concepts used during the dissertation.

- **CPS:** *"physical, biological and engineered systems whose operations are monitored, coordinated, controlled and integrated by a computing and communication core"* [Ran10].
- **ICPS:** A CPSs which is used in an industrial domain such as, Automated Warehouses (AWs) and Press Lines (PLs).
- **Monitoring System:** It is a system able to capture data at runtime from an ICPS that needs to be supervised [JHZ$^+$15].
- **Visualization System:** It is a system able to interpret the data captured by the *monitoring system* in order to convert that data into information through an UI.
- **Supervision:** It is an action that the end user does in order to be aware of what is happening in the ICPS. To do so, it is necessary a *monitoring system* and a *visualization system*.
- **Variability:** Refers to range of possible outcomes of a given situation, e.g., a device can send different (variable) data depending on the situation.
- **Evolution:** Refers to changes that may occur during the operation of the ICPS that cause changes on the output or result of a system (visualization or monitoring system).

- **Adaptive Systems:** Refers to visualizations which change their appearance based on some algorithm. [SCPR06].
- **Adaptable Systems:** Systems which are configured by the user depending on his or her approach or criteria. [SCPR06].

1.4 Research Hypothesis

- **Hypothesis 1:** The use of Model-Based Engineering (MBE) helps creating an unified solution across domains in order to monitor ICPSs. This hypothesis corresponds to challenge 1 (*Managing data acquisition on evolving ICPSs*).
- **Hypothesis 2:** The use of tree models allows to detect and classify ICPS evolution. This hypothesis corresponds to challenge 2 (*Runtime evolution detection and classification*).
- **Hypothesis 3:** Adaptive and Adaptable systems help on decision making when the ICPS evolves. This hypothesis corresponds to challenge 3 (*Personalize user interfaces*).

1.5 Contributions

This thesis aims to contribute the previously identified challenges applying the explained methodology. To respond to those challenges, a solution approach for managing ICPS evolution in monitoring and visualization systems is proposed. The main contributions of this Ph.D. thesis are enumerated below:

- **Modular Monitoring System:** A *Modular Monitoring System* is proposed built by the combination of different standards that are valid for ICPSs with different features. The following guidelines are provided: (I) data representation of the ICPS to be monitored, (II) interaction between the devices and the monitoring system and (III) representation of the captured data in an unified manner. Additionally, the modular monitoring system is able to represent the captured data although the ICPS to supervise has evolved.

- **Personalized Visualization & Evolution Detection System:** An adaptive and adaptable system, i.e., a system capable to (I) detect and classify ICPS evolution automatically by the designed and developed algorithm that compares data structures over time in a standardized format. (II) The system permits the end user to create his/her own visualizations; and (III) considering the visualizations created by the user and the ICPS evolution, the system adapts the UI automatically in order to alert the user about the changes.

1.6 Publications

Different peer-reviewed publications were already presented and discussed in journals and conferences during the Ph.D. studies. The publications that endorse this Thesis are listed below:

- [ILA+17]: *Product Line Engineering of Monitoring Functionality in Industrial Cyber-Physical Systems: A Domain Analysis.* Proceedings of the 21st International Systems and Software Product Line Conference, SPLC 2017, Volume A, Sevilla, Spain, September 25-29, 2017. Pages 195–204. DOI 10.1145/3106195.310622. **Hitachi young best paper award**. Ranking GGS[1]: A-. Related to Chapter 3, Chapter 4 and Chapter 6.

- [IYA+18]: *Model- Based Personalized Visualization System for Monitoring Evolving Industrial Cyber-Physical System.* In 25th Asia-Pacific Software Engineering Conference, APSEC 2018, Nara,Japan, December 4-7, 2018, pages 532–541, 2018. DOI 10.1109/APSEC.2018.00068. Ranking GGS: B. Related to Chapter 5 and Chapter 6.

- [IIL+19a]: *TRILATERAL: Software Product Line based multidomain IoT artifact generation for Industrial CPS.* Proceedings of the 7th International Conference on Model-Driven Engineering and Software Development - Volume 1: MODELSWARD. Pages 64-73. DOI: 10.5220/0007343500640073. Ranking GGS: C. Related to Chapter 4 and Chapter 6.

- [ISA19]: *Industrial Cyber-Physical System Evolution Detection and Alert Generation.* Journal of Applied Science. Volume 9, article number 1586

[1]http://gii-grin-scie-rating.scie.es/ratingSearch.jsf

(2019). DOI: 10.3390/app9081586. Ranking Scopus: Q1. Related to Chapter 5 and Chapter 6.

- [IIL+19b]: *Trilateral: a model-based approach for industrial cps - monitoring and control*. In Communications in Computerand Information Science. Springer International Publishing, 2019. (To be published). Related to Chapter 4 and 6.

1.7 Research Activities

In this section information on who has supported this research besides the research stay are presented.

1.7.1 Research Support

Following a short description of the organizations involved in the development of this dissertation are exposed: IKERLAN, Goi Eskola Politeknikoa and Simula.

Figure 1.2: IKERLAN Technology Research Centre

Figure 1.3: Mondragon Goi Eskola Politeknikoa

Figure 1.4: Simula Research Laboratory

Ikerlan

IKERLAN (Figure 1.2) is a private non-profit Technological Research Centre in the north of Spain. It is the key technological RTD actor within the Mondragon Group, Spain's tenth-largest industrial corporation. Its 300 employees provide advanced technology transfer to industry.

IKERLAN works closely with companies to improve their competitiveness, through the application of technological knowledge in mechatronics, energy and advanced manufacturing. Typical projects involve interdisciplinary work from different knowledge areas to face complex problems. From its creation 1974, IKERLAN has maintained close relations with companies from the machinery and capital goods, domestic appliance, electronics and computing, automotive and energy sectors; where IKERLAN's developments go until the materialisation of final products. As a centre of excellence in the transfer of technology, more than 800 R&D projects were completed so far in cooperation with companies where advanced manufacturing played a central role.

Mondragon Unibertsitatea: Mondragon Goi Eskola Politeknikoa

Mondragon Goi Eskola Politeknikoa (MGEP) (Figure 1.3) is the Higher Polytechnic School of Mondragon University. It is a co-operative integrated into both Mondragon Corporation and Mondragon Unibertsitatea (the University of Mondragon) and is the legal owner of Mondragon Unibertsitatea's Faculty Of Engineering.

The three main activities in this faculty are Teaching Engineering (about 1.800 students), Training to the industry (about 11.000 hours per year) and Research (around 14 M/year, about 17 European projects and more than 40 international agreements, nowadays). Their research model allows them to achieve scientific levels of excellence, integrated with the mid- to long-term needs of their companies. They develop a collaborative strategy with companies, where mutual confidence and objectives multiply the efficiency of resources. In this sense, the technological transfer and innovation continue to be a differential factor of the Higher Polytechnic School of Mondragon.

Simula Research Laboratory

Dedicated to tackling scientific challenges with long-term impact and of genuine importance to real life, Simula Research Laboratory (Simula) (Figure 1.4), Norway offers an environment that emphasizes and promotes basic research. At the same time, they are deeply involved in research education and application-driven innovation and commercialization.

Simula was established as a non-profit, limited company in 2001, and is fully owned by the Norwegian Ministry of Education and Research. Its research is funded through competitive grants from national funding agencies and the EC, research contracts with industry, and a basic allowance from the state.

Its principal activities are training and research, development and innovationn tracts with industry, and a basic allowance from the state.

1.7.2 Research Stay

Visiting other institutions to collaborate with relevant scientist is part of the activities researchers carry out during the Ph.D. studies. Part of the research made in this work was made during a stay in Norway at Simula Research Laboratory in the department of Engineering Complex Software Systems [2]. A total of three months of research stay were carried out (from mid of August 2017 to begin of December 2017), which is one of the requirements for having access to the international Ph.D. mention.

The objective of the stay was to obtain feedback from research experts in the field of CPSs to strengthen collaboration and thus, be able to carry out collaborative research. In addition, given the expertise field of the research group of Simula, it was possible to take advantage of learning more about, software engineering, complex software systems, research methodologies, software or system product lines, etc. As a result of the collaboration two conference papers were published [ILA$^+$17, IYA$^+$18].

1.8 Outline

This section outlines the content of the Thesis. Figure 1.5 illustrates the chapters of this dissertation. Below, a summary of each chapter in this dissertation is provided.

- **II FOUNDATION AND CONTEXT**
  - **Chapter 2**: The background in terms of Industry 4.0 is presented in which the terminology used during the rest of the document is introduced. More specifically, it presents a background on monitoring ICPSs, managing ICPSs evolution and adaptive and adaptable user interfaces.
  - **Chapter 3**: A domain analysis in different industrial domains is fulfilled in order to identify their needs and requirements. Additionally, a Systematic Literature Review (SLR) is performed where primary studies are compared.

[2] https://www.simula.no/research/projects/department-engineering-complex-software-systems

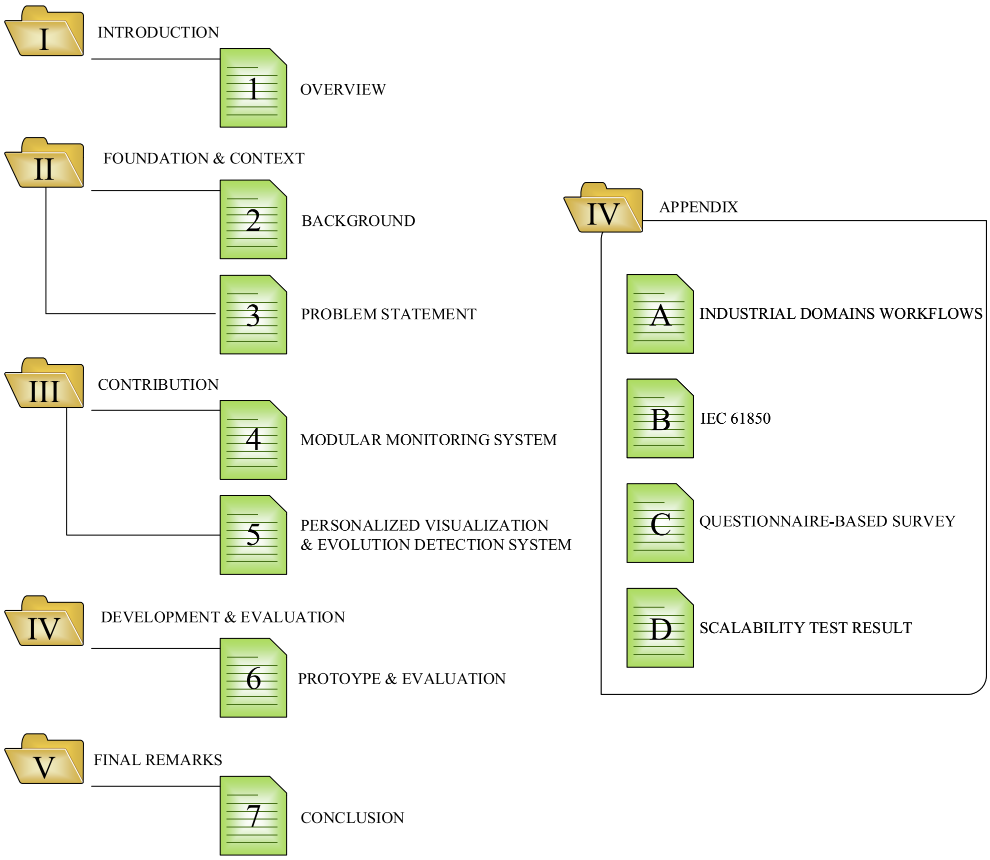


Figure 1.5: Dissertation outline schema

- **III CONTRIBUTION**

  - **Chapter 4**: An overview of the proposed solution followed by the *Modular Monitoring System* where its main components and how the presented system works is presented. More specifically, how a user can configure a concrete monitoring system and after configuring the system automatically starts monitoring the ICPS.

  - **Chapter 5**: The design of the *Personalized Visualization & Evolution Detection System* is presented. First, the main components of the system are presented followed by the process of the system where it is explained how the user is able to create visualizations, how the system is able to detect automatically the ICPS evolution and how these evolution is alerted to the user besides adapting the visualizations automatically.

- **IV PROTOTYPE & EVALUATION**

- **Chapter 6**: A prototype for each proposed system is presented, i.e., the implementation on the *Modular Monitoring System* and the *Personalized Visualization & Evolution Detection System*. Then, the prototypes are evaluated where Return of Investments, Questionnaire-Based Surverys (QBSs) and scalability tests are performed.

- **V CONCLUSION**

  - **Chapter 7**: A summary of the contributions remarking the lessons learned during this dissertation is included. Furthermore, future directions are proposed and discussed.

Part II

Foundation & Context

"Climb mountains not so the world can see you, but so you can see the world."

David McCullough Jr

Chapter 2

Background

In this Chapter[1] the terms of Industry 4.0 and Cyber-Physical System (CPS) are briefly presented. Additionally, considering the main challenges, a background on monitoring evolving Industrial Cyber-Physical Systems (ICPSs), comparison approaches for managing ICPSs evolution and, Adaptive & Adaptable User Interfaces (UIs) for industry are presented.

Contents

2.1 Industry 4.0 & Cyber-Physical Systems **18**

2.2 Monitoring Evolving Industrial Cyber-Physical Systems **23**

2.3 Managing ICPS Evolution: Comparison Approaches **24**

2.4 Adaptive & Adaptable User Interfaces for Industry **25**

[1] The content of this Chapter has been published partially in [IIL+19a, IYA+18, ISA19, ILA+17]

2.1 Industry 4.0 & Cyber-Physical Systems

The industry has received many changes since the 18th century when the first mechanical loom appeared. The second revolution was at the beginning of the 20th, the appearance of first product line. Some years later, in 1970 the 3rd industrial revolution where electronics and Information Technology where introduced in the industry. The 4thindustrial revolution, better known as Industry 4.0 is not a future tendency, it is currently happening. Figure 2.1 shows the four stages of the industrial revolution [KHHW13].

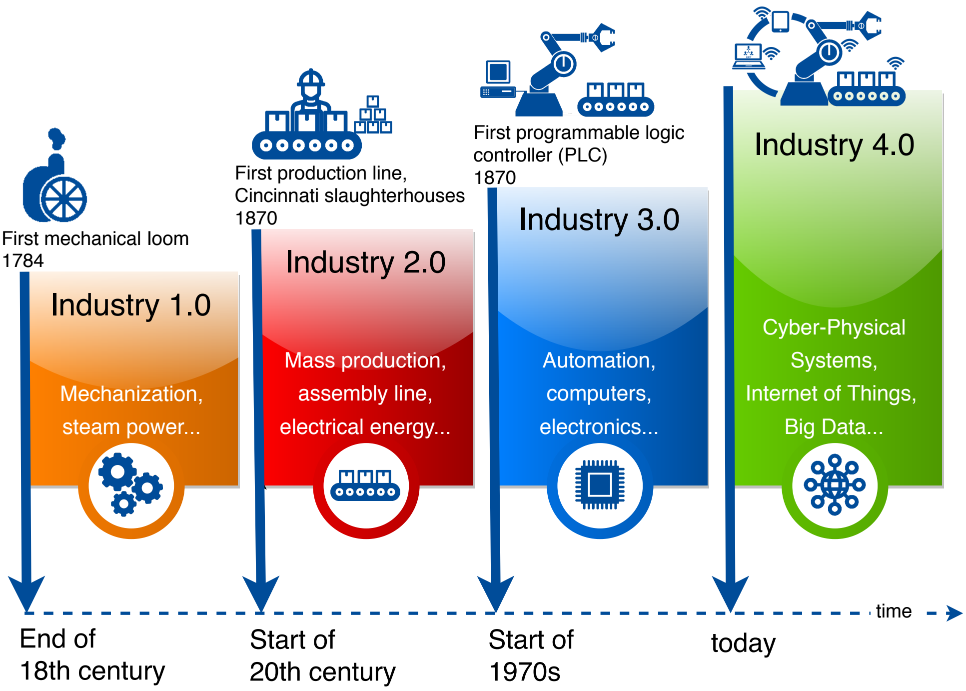


Figure 2.1: Four stages of the Industrial Revolution.

The 4th industrial revolution was coined by Germany' government to describe intelligent factories. Currently, it is being an strategy for Research and Development in many industrial companies. Different companies are changing their technologies to advanced connectivity, Cloud Computing, Internet of Things (IoT), CPS, intelligent processes, etc. In words of PWC [2], in the next five years, great strides are expected on this direction.

Many challenges need to be addressed in this fourth industrial revolution as Kagermann et al. describe [KHHW13]: (I) the importance of considering customer

[2]PWC www.pwc.com

requirements, (II) flexibility in manufacturing systems, (III) provide a right decision making, (IV) making manufacturing more efficient and productive, (V) the importance of analysing data, (VII) collaboration between human and machines, i.e., Human-Computer Interaction (HCI) and also (VIII) better balance between work and life. Additionally, as Lee et al. [LKY14b] conclude that Industry 4.0: (I) can reduce machine downtime predicting machine health, (II) gives the possibility to have more transparent and organized industrial management, (III) gives a better working environment in addition to labor cost reduction, (IV) and will reduce the cost thanks to saving energy, optimizing maintenance and scheduling and supply chain management.

Therefore, Industry 4.0 denotes the trend of industrial technologies, including CPS, IoT and Cloud Computing, which is defined as: *"Industry 4.0 will involve the technical integration of CPS [...] and the use of the Internet of Things and services in industrial processes."* [KHHW13]. The CPSs are the new revolution in the field of Information and Communication Technology, specially in Embedded Systems. They are defined as *"physical, biological and engineered systems whose operations are monitored, coordinated, controlled and integrated by a computing and communication core"* [RLSS10]. A CPS is divided in two parts: (I) the physical and (II) the cybernetics. The former corresponds to the hardware part. The latter one, instead, is the logical part which gives the logic to the hardware besides providing communication capability. Currently the CPSs are used in many sectors, such as, manufacturing, energy, health, intelligent cities and transports [LCK16, SCC+17].

The objective is to provide computation and communication capabilities to the components; in that way, different components can cooperate with each other generating distributed and autonomous ecosystems [LX03]. Therefore, a CPS is more than an individual component which offers services trough the Internet, i.e., IoT. It addresses complete systems which are able to feel and control the physical world in a dynamic way using other systems.

The main advantage of a CPS is the flexibility and adaptation capability, not only on software but also in the inclusion and removal of physical components since all the cybernetic entities are totally decoupled from each other [TFK+18].

The main characteristics of CPSs are [TFK+18, LX03]:

- The capability to interact with physical objects for monitoring and/or controlling a system.

- The use of information saved in the virtual world, which can be use for learning and evolving a system.

In order to get that flexibility, the technical integration of CPSs needs the use of IoT in industrial processes [KHHW13]. The IoT devices deployed in CPS scenarios are embedded devices that usually have more advance requirements in terms of monitoring and control [TZXZ14]. The IoT gives the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction. IoT's main challenge is interoperability. The connection of industrial devices to Information Technology and IoT platforms, can represent business core. There are multiple transport mechanisms in order to connect industrial devices [FBT18].

The IoT includes a set of communication protocols such as REpresentational State Transfer (REST) [PWA13, Fie00], ExtensibleMessaging and Presence Protocol (XMPP) [SA05, 61211], Message Queue Server TelemetryTransport (MQTT) [Loc10], Advanced Message Queuing Protocol (AMQP) [Sta12], Constrained Application Protocol (CoAP) [Che14], Data Distribution Service (DDS) [OMG15], Java Message Service (JMS) [HBS+02], OPC Unified Architecture (OPC UA) [MLD09], etc. Therefore, it is possible to find multiple protocols within this domain: from proprietary solutions to open standards. All of them are maneuvering to be preponderant among the IoT protocols, but it is clear that there will always be a collection of protocols for this type of systems. Therefore, these protocols will coexist, each with its own strengths and weaknesses.

Considering this dissertation is focused on industrial CPSs, such as Automated Warehouses (AWs), Press Lines (PLs) and Catenary-Free Trams (CFTs) where different IoT protocols are need, CPSs in industrial domains is named as ICPSs. In order to develop and integrate CPSs and IoT in different industrial domains [PBM07], two widely-used international standards exist: IEC 62264 (IEC 62264-1:2013) [IEC13b] and its predecessor ISA-95 [ANS05] standard.

The ISA-95 standard defines four automation levels:

- **Level 1** is the lowest level where data is collected. It includes devices that interact with a physical environment.
- **Level 2** controls devices from Level 1 using Supervisory Control And Data Acquisition (SCADA) Systems and Programmable Logic Controllers (PLCs).

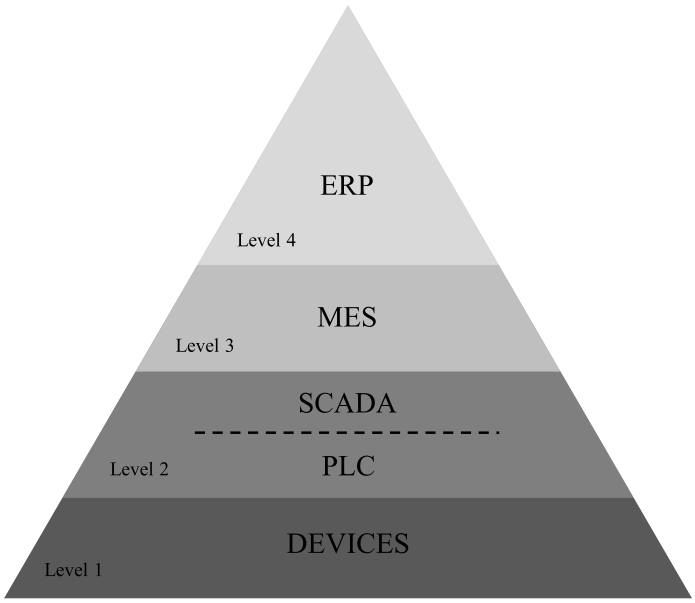


Figure 2.2: ISA-95 standard, four automation levels

- **Level 3** includes Manufacturing Execution Systems (MESs) with key functionalities of manufacturing production and control. MESs also manage and/or control additional operations such as related to maintenance, and quality and/or inventory.

- **Level 4** is the highest level that includes management systems such as Enterprise Resource Planning (ERP), Customer Relationship Management (CRM), and Supply Chain Management (SCM).

In the context of this Ph.D thesis, our focus is up to level three due to the fact that supervising and controlling CPSs in the context of Industry 4.0 is becoming essential.

Regarding International Electrotechnical Commission (IEC) 62264, this standard is part of RAMI 4.0, i.e., the Architectural Model of Industry 4.0 [HB15]. The IEC 62264 describes interfaces between manufacturing control and enterprise functions, i.e., how information can be exchanged in a cost-effective manner in order to preserve the entire system integrity.

Thanks to the IEC-62264, the interfaces between the business systems of an enterprise and its workflows activities can be defined. For that, as shown in Figure 2.3 (left), a role-based equipment hierarchy is defined:

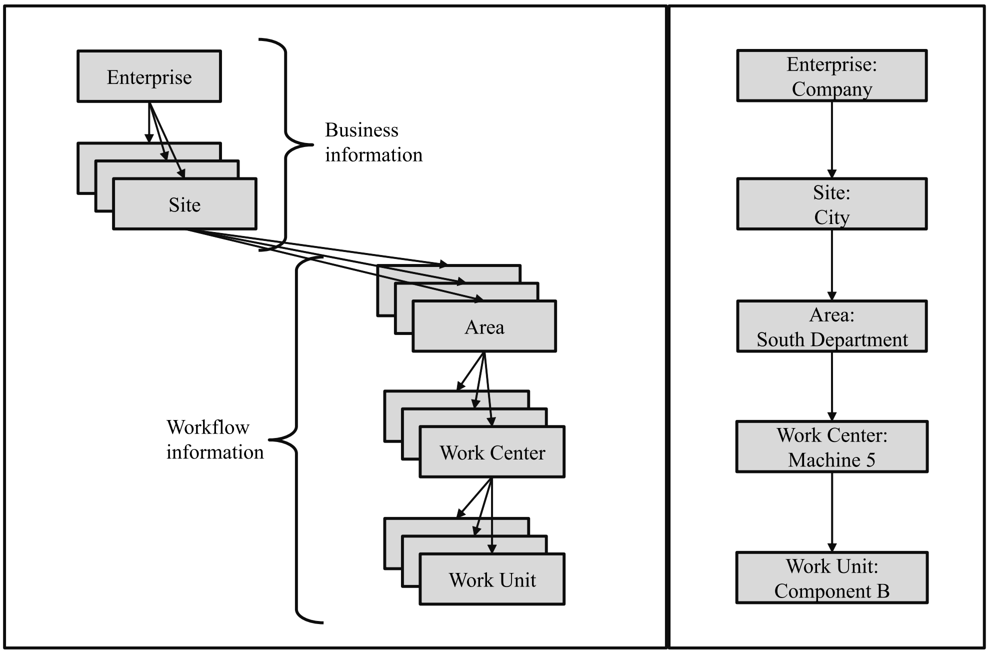


Figure 2.3: IEC 62264 standard: Left) The information model. Right) Information model example [IEC13b]

- **Enterprise**: This is the top level of the role-based equipment. It captures information about the industrial domain to be controlled or monitored.
- **Site**: This determines a physical, geographical or logical grouping of the enterprise to be controlled or monitored.
- **Area**: This is the physical, geographical or logical grouping of the Site. It is used to identify the elements inside Site.
- **Work center**: These are equipment elements located under the Area which reflects the operation management.
- **Work unit**: The work unit gives more concrete information about the work center.

Thanks to this standard it is possible to organize an ICPS inside an industrial domain in order to know the relation between all the devices within an ICPS.

2.2 Monitoring Evolving Industrial Cyber-Physical Systems

Monitoring enables the transition of a traditional manufacturing system, towards an ICPS in the context of Industry 4.0 [KHHW13]. It is essential to monitor system processes for industrial domains to detect anomalies and avoid system shut-downs with the ultimate aim of improving productivity and reducing cost.

When an ICPS is monitored, data from large amounts of heterogeneous, distributed and autonomous devices are received [LKY14b, LBK15]. In addition, every ICPS has a different composition, i.e., it is composed by different devices which depends on every use case [ILA+17]. The objective of monitoring an ICPS is to process the data (received from an ICPS) into information. A monitoring system can lead to improvements in productivity [LKY14b]. As Fleischmann et al. expose [FKF+16], a monitoring system helps to identify errors at early stage and it is indispensable for generating product successfully. Additionally, by analyzing these data, anomalies during operation can be identified [BBA+17, SDSS16].

There are some proposals in the literature that address monitoring solutions. Some authors propose monitoring ICPSs to detect attacks that can affect the systems [PDB13] or even for, storage data, data analysis and use of machine learning techniques to automatically update ICPS functionalities [NBK+15]. Chen et al. [CZH17] present a mobile phone manufacturing enterprise where different devices are monitored in order to make decisions. The authors are able to identify each device when this one connects with the monitoring system in order to communicate any alert to the end user. Others present a chemicals monitoring system that can accurately obtain monitoring data and alert about anomalies which does not meet already defined requisites [Hu16]. Nuclear radiation environments are also monitored at real-time in order to control the radiation in the environment [LHZ+16], or even to control workers work [TFPE12].

Even if monitoring systems are used for different proposes and in different domains such as traffic control and safety, manufacturing or energy conservation [Lee08], all of them need to collect, display and analyze data [SHSH09]. The problem arises when the system to be monitored evolves due to system maintenance, retrofitting, resource optimization or because the system needs to be adapted to new services [GPM14, GBH+16, LCK16]. As Gerostathopoulos et al. expose [GBH+16] “CPSs are becoming more modular, dynamic, networked, and large-scale".

Thus, in an ICPS the cyber part or the physical part can evolve [CZH17]. Considering this, the monitoring system associated to the ICPS needs to be adapted. Therefore, flexibility and adaptation are the two required capabilities posing additional challenges for monitoring ICPSs [GPM14, GBH+16, LCK16]. Furthermore, there are no proposals in the analyzed contributions that manages the impact on the monitoring system when the ICPS evolves.

2.3 Managing ICPS Evolution: Comparison Approaches

Considering that ICPSs evolve over time, detecting evolution is important, otherwise being aware of what is happening in the ICPS is not possible. In other domains, when evolution or changes need to be detected tree-shaped model structures are used, e.g., in medicine, tree models (mutation tree) are used to identify the mutation of atoms, molecules, particle, etc. [CLW+16]. With a tree-shaped models, the structure beside the information is transferred [WSBR12]. Other authors, such as Melnik et al. use the comparison of models to then match the two trees, in order to turn them into one, but the author does not make emphasis on detecting which the evolution is [MGR02]. Thus, it seems to be usual to use tree models in order to manage evolution.

In order to detect evolution by comparing tree models, different comparison approaches exist in the literature [SC12]: code comparison techniques, textual model comparison [CAM02], code clone detection and translation to formalism.

Table 2.1: Comparison between different comparison solutions

Solution	Graphical Output	No Graphical Output	High Level	Low Level	Runtime Use	Characteristics	Comments
SiDiff[3]		■	■			similiratiy based	
[XS05]	■		■			object-oriented	
WinDiff[4]	■		■		■	line-based	don't detect edits
WinMerge[5]	■		■		■	line-based	don't detect edits
Javers[6]		■		■	■	object-oriented	
[CAM02]		■		■		ad-hoc	
[WSBR12]		■				ad-hoc	
Hibernate[7]		■		■	■	object-oriented	

As shown in Table 2.1, different methods for the comparison exist, which are used for tree-shaped models [SS08] such as SiDiff, WinDiff, and WinMerge. Some are text comparators like SiDiff and others are graphic comparators (UM-LDiff, WinDiff, and WinMerge). Although the graphics show the result visually, they are not suited for runtime use and our need is to find such changes at runtime since they are manual tools. WinDiff and WinMerge are line-based tools, i.e., changes will not be expressed at logic issues. Trip-wire[8] is able to detect changes but it is not able to detect which is the change and Remedy [9] is able to detect changes in the structure but not on the values.

Cobena et al. and Weaver et al. have selected ad-hoc solutions in order to analyze the whole architecture, i.e., the ICPS structure and the corresponding values. The former one proposes an algorithm to compares XML files [CAM02]. The solution is for web sites and the movements inside the same father are considered changes. The latter instead [WSBR12], even if they are able to detect a change in a concrete moment in time, do not retain the traceability of changes. In addition, the output script must be saved and processed for it to be interpretable. Thus, four different comparison solutions exist, similarity based, objected-oriented, line-based and finally there also exist ad-hoc solutions. Depending on the interest of the comparison, one solution or another may be usable, but considering the analyzed contributions none of them has been used for the detection of ICPS evolution.

2.4 Adaptive & Adaptable User Interfaces for Industry

After capturing data from the ICPS, it is necessary to visualize the information for decision-making [ABY14a]. A good visualization helps a user to make data comprehension and interpretation easier, facilitate effective communication among colleagues, and facilitate improvements [JN11, dR17]. That is why, guides, which guide users when data needs to be presented, exist [Abe13].

Kennard and Leangey [KL10] confirm that the 50% of the time used for developing an application is used for developing the UI and the 48% of the code is front-end code. Thus, half of the time inverted in addition to the half of the code generated is in the visualization [KL10, MCS14]. Thus, semi-automatic UI generation and

[8] https://github.com/Tripwire/tripwire-open-source

[9] http://www.bmcsoftware.es/it-solutions/remedy-itsm.html

UI code reutilization is as important as semi-automatic back-end code generation and reutilization. Delgado et al. [DEJE16] affirm that in many industrial cases the productivity besides the software quality increases when a software is reused. In terms of visualization they concluded that reusing UI assets provides significant benefits, reducing cost-consuming tasks between 34% and 75% depending on the project.

Considering that and taking into account ICPS can evolve time to time, apart from adapting the monitoring system, the corresponding visualization needs to be adapted to the end user when the ICPS evolves. But as Akiki et al. describe [ABY14a]: “The interface should adapt to the user; rather than the user adapting to the system”. Otherwise, the user needs to make a bigger effort for marking-conclusions. Thus, there exist for facing UIs, i.e., the development cost [SPHV10, KL10, MCS14] in addition to the diversity of users and the evolution and the maintenance of the visualization [SPHV10] are terms to take into account.

Inside the UI adaptation two different concepts need to be described: adaptability and adaptivity [MCS14]. The former one is an adaptation where the information for adapting the UI is available prior to the interaction. The latter one instead, it is a self-adaptation where the information is acquired during the interaction. Additionally, other classification exist, adaptive and adaptable UI generation [SCPR06]. An adaptive UI change the appearance of an UI based on some algorithm, instead, adaptable UI, is configured by the user, depending on his or her approach or criteria.

In terms of adaptable UIs, Lehmann et al. presented different studies about models manipulation or execution at runtime for UIs [LBFA08, LBA10]. In the same manner, [GFSV14] presented a collaborative modelling environment. Despite being systems capable of managing runtime changes, in both cases it is the developer who decide how to visualize data. Delgado et al. also presents a similar runtime adaptive system [DEJE16]. Others proposed adaptable UIs, using WebRatio in cooperation with Interfaction Flow Modeling Language [ABBB15]. As well as Schramm et al., [SPHV10] that research about automatic UIs generation using model drive development. As for adaptive UIs, Mezthoudi et al. [MMKV15] presented a runtime UIs generation using models. The author presents two different adaptations in a practical use case, i.e., car rental web page. The first one depends on users' culture, thus depending on it, the UIs changes. In the second one, instead, the visualization will be different depending on the platform where is going to be visualized. Other authors present adaptive web pages for smart home environments [BLA10]. Considering the different contributions analyzed, uncertainty is not something common, i.e., the adaptive systems, do not consider unexpected changes in their visualizations during runtime.

Table 2.2: Comparison between Adaptive and Adaptable UIs

	Runtime adaptation	**Adaptable UI**		**Adaptive UI**	**Unknown data**	**Models**
		developer	**user**			
[SPHV10]		■			■	■
[LBFA08]	■	■				■
[DEJE16]	■	■			■	■
[LBA10]	■			■		■
[GFSV14]		■				■
[BLA10]	■			■		■
[ABBB15]		■				■
[SCPR06]	■		■			
[MMKV15]	■			■		■

In Table 2.2, different case studies or practices are presented about UI semi-automatic generation and reutilization. Notice that the majority of the case studies are generated using models and most of them consider runtime adaptation. Additionally, adaptive and adaptable UIs are concepts which are not usually combined. Usually the developers are the ones who adapt the UI to the end user instead of the end user him/herself and in general, not designed features are not considered in the visualization. Additionally, not many authors manage systems able to receive data that was not received before (unknown data).

Considering the analyzed contributions, the unique study presented that considers adaptable UI by the user in industrial settings was presented in 2006 by Stürzlinger et al. [SCPR06]. They presented a tool (Façade), capable of changing existing UI of different applications inside the computer using drag-and-drop paradigm. It is an application capable of taking any component from a UI and moving to another place, but, the application does not study unknown features. Thus, in the analyzed literature neither has give the chance to the end user in industry to create his/her own visualization, nor adapting the necessary visualizations and alerts at runtime when the ICPS that is being monitored evolves.

Chapter 3

Problem Statement

This Chapter[1] presents the problem statement of this Ph.D thesis. First, the chapter is introduced explaining the process followed to perform the domain analysis. Then, the analysis of different industrial domains is exposed. Finally, a Systematic Literature Review (SLR) is performed in order to analyze if the identified requirements are addressed in the literature.

Contents

3.1 Introduction **30**
3.2 Domain Analysis **32**
3.2.1 Industrial Domains Information 32
3.2.2 Domain Analysis Outcomes 42
3.3 Revisiting Requirements through a Systematic Literature Review **47**
3.3.1 Process for a Systematic Evaluation of the Primary Studies 48
3.3.2 Results of the Systematic Literature Review 52

[1] The content of this Chapter has been published partially in [ILA+17, IIL+19a]

3.1 Introduction

Bearing in mind the fourth industrial revolution, it is necessary to monitor and control industrial systems. Therefore, to gain insights on those problems and provide inspiration for potential solutions, a domain analysis is conducted for three domains , i.e., Automated Warehouse (AW), Press Line (PL) and Catenary-Free Tram (CFT) which belong to different vertical sectors, such as logistics, manufacturing and transport. Particularly, the main objective of conducting a domain analysis is to identify common requirements when an Industrial Cyber-Physical System (ICPS) need to be monitored and the information visualized.

The main goal of the domain analysis is to identify if any commonalities and variabilities exist in terms of supervising ICPSs in this three different domains. Thus, as a starting step towards achieving different industrial domains needs, the following objectives for conducting the domain analysis are defined:

- Understand each industrial domain to identify the main requirements and needs for supervising their ICPSs.
- Analyze the industrial domain in order to be able to satisfy the identified requirements for supervising their ICPSs.
- Identify commonalities and varibilities among the industrial domains regarding monitoring and visualization systems in Industry 4.0.

To conduct the domain analysis, the process of Figure 3.1 was followed. First, each domain was analyzed independently with the technical manager of each industrial domain via interviews and Questionnaire-Based Surverys (QBSs) (see Appendix C.1). The interviews were informal meetings where the technical manager had the chance to speak about their daily work, interest, concerns, etc. Thus, although the flow of the conversation was guided, it was the technical manager who led the interview. The interviews were carried out to get closer information, while the questionnaire was conducted to gather more technical information. With interviews and questionnaires, the following information from each industrial domain was collected: (I) product and domain needs and requirements, (II) current industrial domain characterization, (III) current working process and daily work information and (IV) technical manager' challenges and expectations.

Once all individual information about each industrial domain was collected (i.e., *Industrial Domain Information*), a report was written. It should be noted that as the interviews were guided, almost the same type of information was retrieved for each domain. Through guided interviews, it was possible to gather information in terms of commonalities and variabilities across domains when an ICPS needs to be supervised by an end user. If exist, requirements in order to give a common solution within or across the domains would identified, but always taking into account the existing variability.

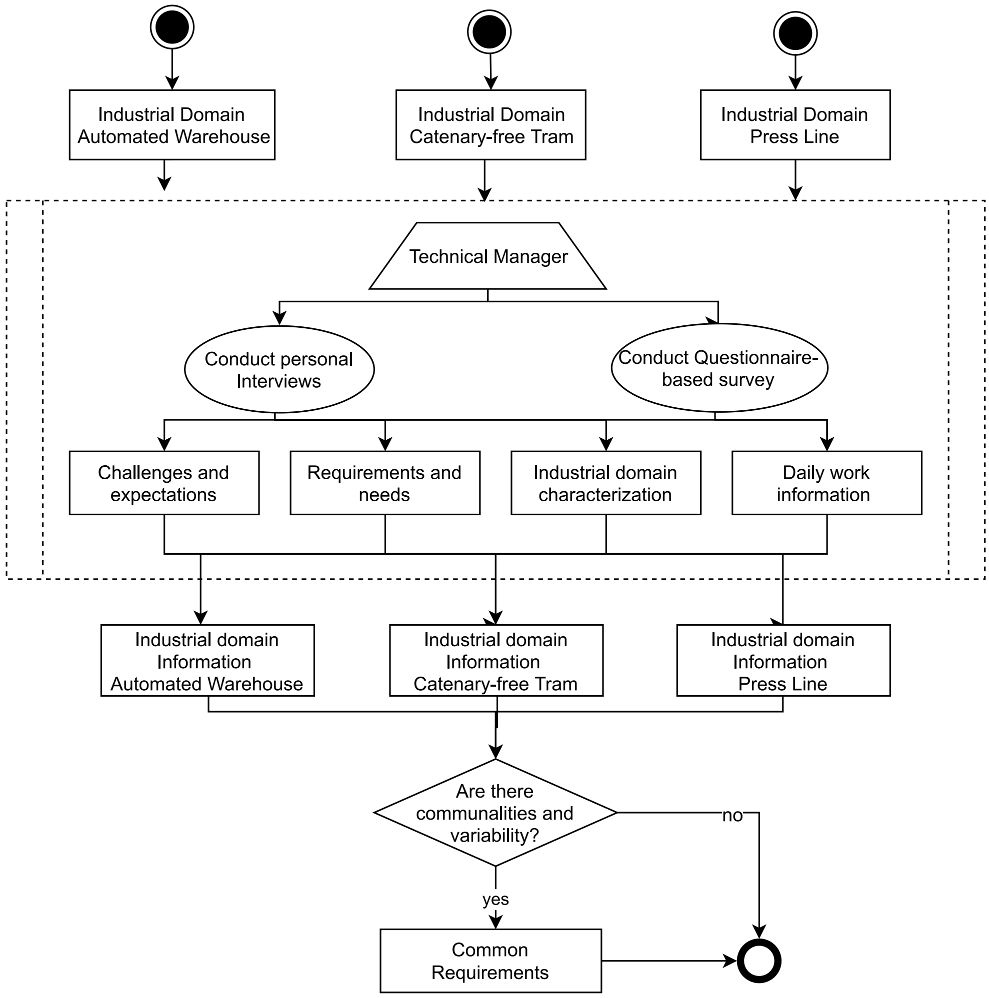


Figure 3.1: Domain Analysis Process

3.2 Domain Analysis

In this section (Section 3.2.1) each industrial domain is explained and analyzed. Then, in Section 3.2.2the common requirements, i.e., the requirements identified within and across the different industrial domains, are presented.

3.2.1 Industrial Domains Information

During the domain analysis, the importance of supervising every ICPS is identified. For this propose, data needs to be monitored and the gathered information needs to be visualized. Other requirements were also identified when an ICPS needs to be supervised by the end user: existing variability, evolution and the existing user roles. In this section the information collected from each industrial domain is presented. The information is structured based on these three requirements.

Automated Warehouse

ULMA Handling Systems[2] has its main headquarter located in Oñati (Spain) and belongs to the 10th largest industrial corporation in Spain, the Mondragon Group[3]. ULMA Handling Systems develops automated handling systems for AWs all over the world in different sectors such as food, industrial, textile, and storage. The company's engineers design, produce, install and maintain material handling systems in facilities ranging from small AWs to much more complex ones. An AW is a physically distributed system that is composed of autonomous devices carrying out industrial automation tasks, such as transferring, and sorting [MDCTS14]. The main objective of an AW is to store and retrieve goods with minimal human intervention [ABCC05]. A typical workflow of an AW is explained in Appendix A in Section A.1.

Considering the AW is composed by autonomous devices, monitoring the different devices in order to supervise the real ICPS is important. In addition, each device can be able to make decisions which makes the domain complex. Thanks to monitoring different devices and visualizing that information, it is possible supervise the ICPS, e.g., the position, the weight, etc. of each good within the AW. Therefore, the end user

[2] https://www.ulmahandling.com/en/
[3] https://www.mondragon-corporation.com/en/

is aware of the real state of the AW and can make decisions avoiding loss of money or productivity.

Variability. AWs are classified into three groups: small, medium and large, depending on the quantity of devices (see Table 3.1). Different customers have varied types of AWs that require to be customized in the design as well as in the development, e.g., variations in the plant layouts and the behaviour of the devices within an AW. An example of a (partial) layout of a real AW is shown in Figure 3.2. An AW is organized in different groups, which are classified either into logical or physical depending on customers' requirements. Each group contains different devices, which are grouped into different device types such as scanners and tables. Additionally, each device is able to send different attributes or values depending on the state of the AW.

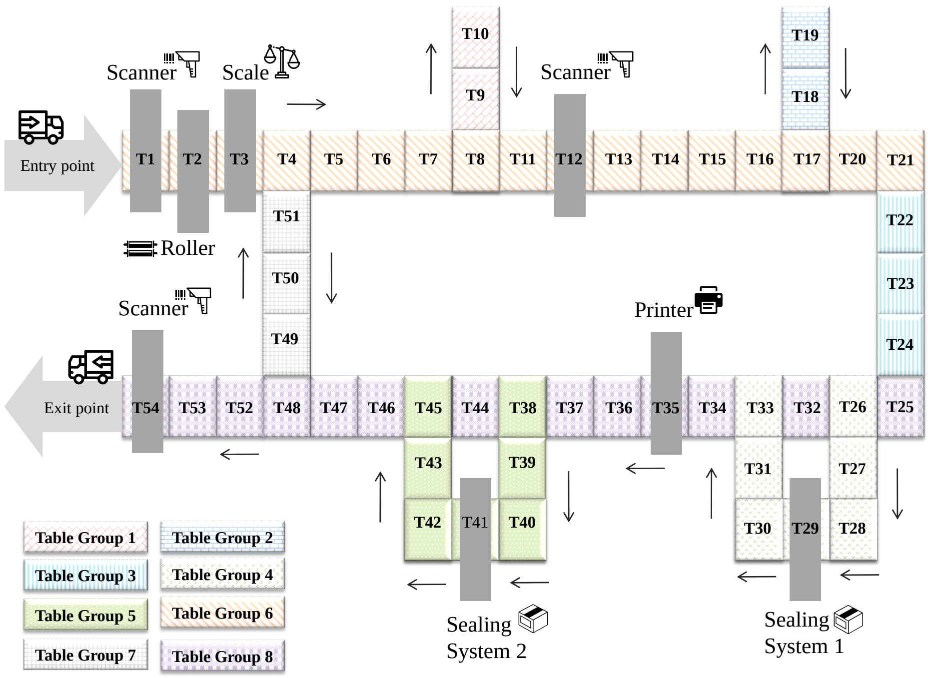


Figure 3.2: Example of an Automated Warehouse layout (Partial)

The small warehouses require to control and monitor at least 50 devices and the large ones more than 500. Similarly, the number of anomalies (malfunctions of the system) per week is related with the number of devices installed in an ICPS, i.e., the more devices, the more anomalies. An incidence can be detected and resolved automatically by the system itself, e.g., by restarting the associated activity, even though the cause of the incidence is unknown. Incidences are prioritized based on the occurrence frequency, and critical problems are addressed during the same working day.

Table 3.1: Characteristics of the Automated Warehouses

AWs Characteristics	**Product Scale**		
	Small	**Medium**	**Large**
Average number of devices	50 to 99	100 to 500	>500
Incidences per week	1 to 9	10 to 29	30 to 50
How many days are needed to address issues that caused incidences	1 to 2	3 to 5	5 to10
Percentage of ICPS affected when the ICPS evolves (added, removed, replaced devices)?	80% to 100%	60% to 79%	40% to 59%
Users supporting ICPS	1 to 9	1 to 9	10 to 29

Evolution. Furthermore, an AW can evolve, either due to the incidents that occur during operation or due to customer requirements changes. In Table 3.1 technical information about the AWs is exposed. In small AWs between 80% to the 100% percent of the system is affected, instead in big ones the 40% and 59% percent. This may be because even if in big AW the number of affected devices by evolution is higher, the affected area should be small.

The introduction, removal or modification of devices is caused by different reasons, e.g., a device needs to be replaced because it has been damaged, the same model is not available on the market; or new devices, type of devices or even new groups need to be introduced to expand the warehouse to increase its capability; or existing devices need to be replaced with new ones because they provide enhanced features (less energy consumption, more reliability, etc.).

User Roles. It is important to note that different people can be supervising the AW. Table 3.2 shows the different user roles. Not all of them need to visualize the same information, hence, it is necessary to adapt the visualization to the end user. For example, the operator focuses on the runtime data, instead, the warehouse manager, analyzes historical data.

Table 3.2: User Roles for Automated Warehouse Supervision

Role	Definition
Operator	It is the group of people who control the operation of the AW.
Warehouse Manager	It is the person or group of people responsible to analyze the historical data of the ICPS that is being monitored.
Domain Expert	The responsible of analyzing at runtime the raw data of a specific device or group of devices to detect any malfunction or anomaly.
Maintainer	They verify if exist any anomaly in the system, and perform corrective actions if it is necessary. They are also responsible to fix the incidences.

Catenary-Free Tram

In the transport domain, there are companies who focuses on the design, development and commissioning of trains and trams. Trams are a means of passenger transport that runs on rails and in urban areas. This specific use case is focused on a type of tram, i.e., CFT, which is a special type of tram being used increasingly in Europe. Its great advantage compared to a normal tram is that it does not need a catenary. As a result, the infrastructure necessary for its installation is simpler. Additionally, in order to convert the CFT in a controlled environment for the citizen's safety, is important to monitor different devices within the CFT, e.g., engine and brakes status, batteries health, etc.

In the CFT domain, two different peculiarities need to be taken into account. The former is when the tram is ongoing, the environment is mobile and can have power or connectivity limitations. The later is the tram can bulk much more data when it is on a station or a stop. However, since a CFT has to follow a schedule, the time for bulking the data is limited, so the data transfer time must be taken into account. Thus, depending on whether the tram is on going or in station the communication protocol to use will be different. Note that the quantity of information to bulk, the speed, etc will be different thus, it is necessary to adapt the system to multiple scenarios.

Variability. Depending on customers' needs the quantity of wagons will be different and each wagon has its own characteristics as shown in Figure 3.3.

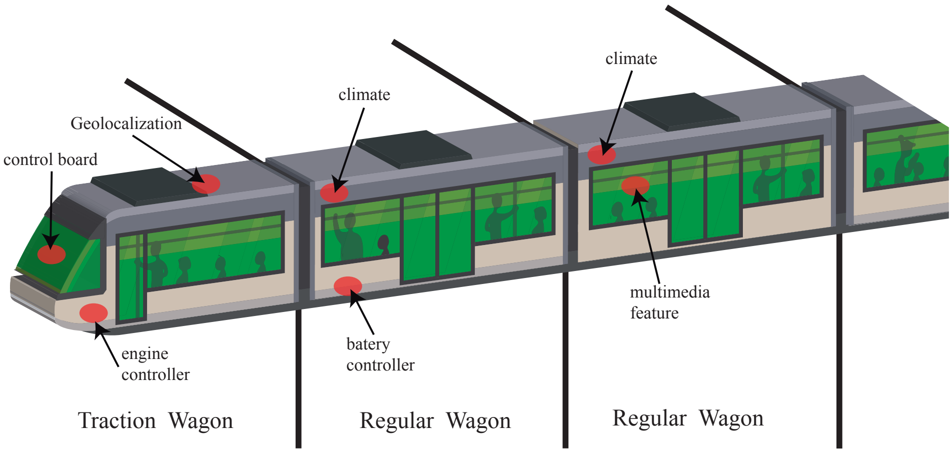


Figure 3.3: Catenary-Free Tram partial Layout

Thus, any transportation system has different parameters to monitor its state, from critical parameters such as speed, direction or maintenance related information (e.g., state of the power source, break wear, etc.) to non critical systems such as information or multimedia features, or climate systems.

In a regular wagon usually a battery controller is funded, i.e., the battery controller is the component that controls the entire energy system to make the tram move from one place to another. This is composed of around twelve different devices that are capable of sending more than 200 attributes or values. In addition to the energy manager, there are other components such as speed and resistance profiles, engine controller, braking motors, auxiliary loads, etc.

Therefore all CFTs are not equal, i.e., CFTs have different features depending on the city to be installed, in the mechanism used, the route to be followed, etc. Because of that, each train will depend on the needs and requirements of the customer. Note that this kind of trams work without catenary, this means that they need to collect energy to move forward and for this, different techniques can be used. For example, in some kind of trams fast change accumulators are used, instead, other CFTs get energy from the rails themselves. Thus, even if both of them are related with energy, depending on the system, the devices within the CFT are different. Non critical systems can also be installed in order to collect more information about the tram (e.g. multimedia feature), but these depend on the customers' requirements.

Evolution. Over time, elements that were considered non-critical can be considered critical, which can provoke to make updates in the devices in order to have a mayor control about them. Additionally, time to time the tram can evolve, e.g., the customer may want to introduce more devices into their tram in order to control more their CFT or a new wagon needs to be introduced to give support to more people. Thus, the structure of a tram can evolve time to time. Furthermore, the devices can evolve, i.e., they can be damaged and therefore must be changed or they may even become obsolete over time and require replacement.

User Roles. Regarding the necessary support for the control and/or supervision of a CFT, four different roles exist. In the Table 3.3 the different roles and their definition are shown. This means that each role works with a different type of information and that the visualizations must be different.

Table 3.3: Catenary-free Tram User Roles

Role	**Definition**
Operator	It is the group of people who control the operation of the CFT.
Signaling Controller	They are responsible for controlling all the signaling in order to control all the movements of the CFT.
Maintenance	They verify if exist any anomaly in the system, and they make decisions if it is necessary.
Vehicle Controller	The person or group of people who operate the machine inside the CFT. They are responsible for controlling all the parameters inside the CFT.

Press Line

Fagor Arrasate located in Arrasate (Spain) is a world leader in metal forming and press machine. The company designs and manufactures mechanical and hydraulic press machines, complete stamping systems, transfer presses, robotic lines, etc. Manufacturing production lines designed and developed by Fagor Arrasate are based on press machines and this dissertation refers to them as PLs. Thus, each PL is different depending on customer needs. In order to be aware of what is happening in the real industrial domain, it is necessary to control every component within the PL through devices installed. Capturing data from the devices can give information about the PL, e.g., information on the mechanism of the machines (the temperature, speed, position, etc). These makes possible to control the real environment in order to avoid bad decision than can cause adverse effects in the production and therefore in the company. Notice that each device is different since can have each own logic, i.e., they can make decisions, making the domain complex.

Variability. Notice that different types of PLs exist (press hardening lines, composites forming lines, pickling lines, etc.). The quantity and type of machines that constitute a PL are different. For example, a hot forming manufacturing line for boron steels (Hot Stamping of Boron Steels) contains 3 fundamental machines, one tied to another:

- **Destacker**: It is the robot responsible of :(I) unstacking previously cut formats and (II) introducing the format in the furnace.
- **Furnace**: Inside this machine, the material remains for a minimum time until reaching a completely austenitic structure, and finally achieves the diffusion of the coating in the substrate. Currently, in Fagor Arrasate PLs, different furnace types are used: (I) roller furnaces, (II) multilevel furnaces, (III) furnace "carousel".
- **Press Machine**: Once the format is heated, the press machine changes the shape of a workpiece with pressure. The main characteristic of this machine is that, unlike the trajectory that is necessary in the forming of cold steels, in the Hot Stamping, the press has to approach the mold as quickly as possible.

In turn, each machine within the PL is composed of different devices which are also variable. One of the main component of a PL is the press machine. As shown in Figure 3.4 (Left), a press machine has two rigid platforms (i.e., head and base), a bed, a ram, and a mechanism. The mechanism is responsible for moving the ram, which will set the base mechanically, hydraulically or manually, as specified by the mechanism. Figure 3.4 (Right[4]) illustrates some sensors that a typical press machine has, such as, thermometers, pressure transducers, and flow meters. A typical workflow of a press machine is explained in Appendix A in Section A.2.

[4]http://www.goizper.com/

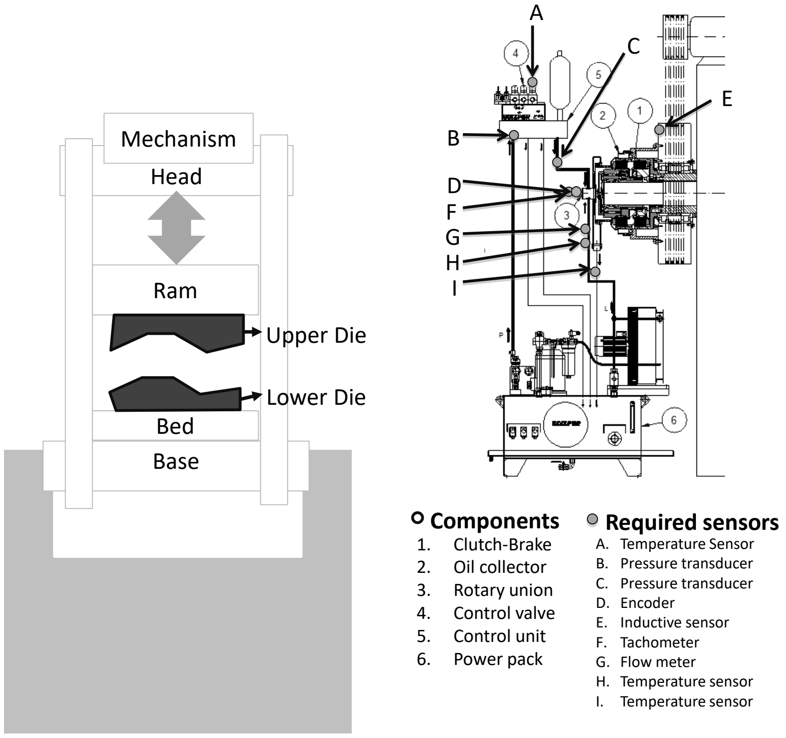


Figure 3.4: Press Machine composition: Left) Main components. Right) Specific layout example

Additionally, three different sizes of press machines can be used inside a PL: large, medium, and small (see Table. 3.4). Though the number of devices is incremental to the size of the machine, the incidences (i.e., machine breakdowns) occurring per week are similar in all press machine sizes and are resolved in 1 to 2 days. For example, the press machine used in the Crema European Project can produce ten workpieces per minute on average indicating that a stroke happens every 6 seconds[5]. If the machine is not correctly configured and the final work is rejected, a significant monetary loss will happen. Thus, the machine has to be correctly configured before the final work is produced and an anomaly in the machine configuration must be detected as soon as possible.

In the next table, information about a press machines is exposed, but notice that a PL is composed by different machines, thus, the quantity of devices in an ICPS is higher than the one presented.

Each machine within a PL has a different objective and therefore, the charac-

[5]CREMA: Cloud-based Rapid Elastic MAnufacturing: http://www.crema-c2networkshop.com/

Table 3.4: Characteristics of the Press Machines

Press Machines Characteristics	Product Scale		
	Small	**Medium**	**Large**
Average number of devices	20 to 49	50 to 99	>500
Incidences per week	>50	>50	>50
How many days are needed to address issues that caused incidences?	1 to 2	1 to 2	1 to 2
Percentage of press machine affected when the machine evolves (added, removed, replaced devices)?	40% to 59%	40% to 59%	40% to 59%

teristics of each one are different. Additionally, note that each machine (e.g., press machine) can be different, e.g., devices can be from different providers, the mechanism of the machine can be different, etc.

Evolution. Furthermore, for example, the automotive world is a sector that is in constant movement and where technological developments require a continuous technological renovation. For example, the Hot Stamping of Boron Steels, a recent creation technology that is settling in the sector and which processes are in constant evolution, change and improvement.

In addition, each machine within in a PL due to machine maintenance, retrofitting, etc., can evolve since each machine is independent. Note that an evolution can affect between 40% and 59% of the machine (see last row of Table. 3.4). New requirements usually have an impact on the devices inside each machine, having to insert new devices (e.g., new elements of an industrial domain need to be monitored due to customer requirements so new devices must be inserted), remove existing ones (e.g., due to an anomaly the device is damaged and must be removed) or modify them (e.g., a device is updated and is now able to send more data that was not previously considered).

Thus, devices inside the machines can evolve, besides, the PL itself (e.g., a new machines can be introduced). But it is important to remark that the Overall Equipment

Efficiency (OEE) should be maximized; and from the evolution perspective, it implies to reduce the (un)planned machine stops, since that would bring negative consequences to the production (e.g., loss of money).

User Role. In this particular use case (i.e., Hot Stamping of Boron Steels), between 30 and 50 people are needed to support the proper functioning of the PL. Notice that the quantity of people would depend on each PL to supervise.

Therefore, being aware of what is happening in the industrial domain is crucial, when an evolution happens many people need to be alerted in order to detect anomalies to reduce system downtime. In Table 3.5 the different roles that the users have in order to support a PL are shown. Thus, depending on the user role, the interest of the users in terms of data is different. In spite of that, all of them need to be aware of what is happening in the PL that is being supervised.

Table 3.5: User Roles for Press Machine Supervision

Role	**Definition**
Operator	It is the group of people who control the operation of the press machine.
Analytical Manager	The person who analyze the historical data in order to to find machine patterns or trends.
Domain Expert	The responsible of analyzing at runtime the raw data of a specific device or group of devices to detect any malfunction or anomaly.
Technical Assistant	They are people that provide technical assistance, i.e., it is the group of people in charge to solve any incidence that can occurred as fast as possible.
Assistance Management	If an incident cannot be solved by the Technical Assistants, a more exhaustive assistance has to be planned. Thus, in that case the issue will be transferred to the Assistance Management in order to solve the incidence.

3.2.2 Domain Analysis Outcomes

After analyzing the different industrial domains, it is possible to notice that common requirements exist across and within different domains, i.e., AW, PL and CFT:

- **Monitoring Necessities:** In all three domains monitoring is necessary. In AW domain is necessary to monitor & control their ICPSs in order to see if all goods follow the right path, i.e., if the goods are successfully stored and retrieved. In the same manner, the CFT domain needs to be controlled & monitored to avoid any problem within the city. Likewise, in PLs in order to avoid any erroneous final works is necessary to monitor the machines. Otherwise the company can incur in a monetary loss.

  To be more specific, e.g., in AW domain, in order to store an retrieve goods, it is necessary to control where, how, etc., are the goods in every moment. Note this is an autonomous system which requires minimal human intervention, hence, it is necessary to supervise it in case any anomaly occurs or even to prevent anomalies. An unplanned stop of the ICPS can cause losses of money being this a negative issue for the company. The same applies in both PL and CFT domains, a stop of the system has negative consequences. It is therefore necessary to be aware of it while the ICPS is on operation. As for the CFT domain, the monitoring of each CFT is necessary for safety reasons. This is why, an anomaly in a CFT, especially in critical elements, can cause problems in which citizens are involved. Therefore, more important than the loss of money that a CFT can cause if it is out of service, is the implication of people, making necessary to develop safety Integrity Level compliant systems.

  Thus, a monitoring system is necessary in order of capture the state of the ICPS during operation with the ultimate goal of detecting anomalies, improving productivity and avoiding system downtime. The importance of monitoring system is also recognized in the literature [KHHW13, CZH17, SHSH09].

- **Similar ICPS Structure:** Analyzing different domains, it is noted that the ICPS structure is similar. In the case of warehouses (see Figure 3.5A), these are distributed over different floors (e.g. Floor 1). In turn these floors are divided into several groups, e.g., printing area, storage area. Each group is made up of different devices (e.g. printer, scale) and these are responsible for sending information to the monitoring system. Therefore, each device may have attributes in which it sends specific information (e.g., code, time, weight).

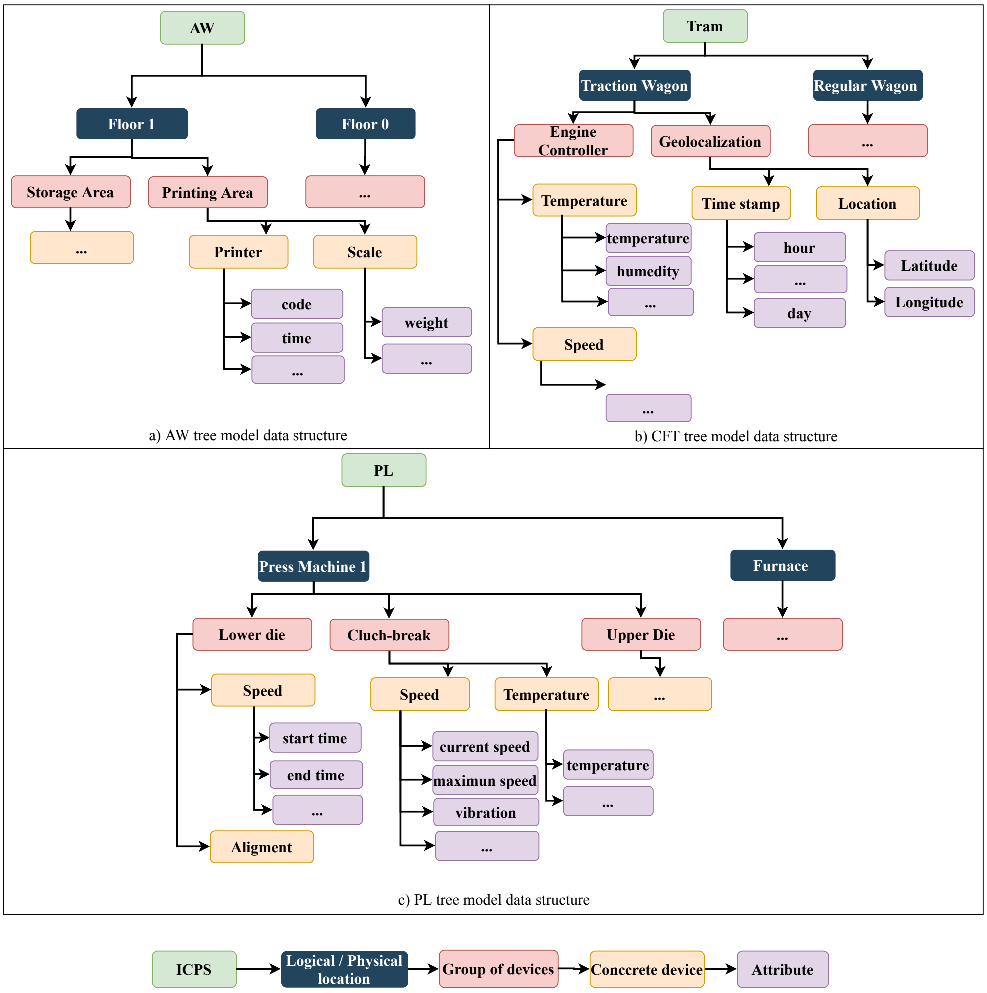


Figure 3.5: Similar data Structures across the domains

A similar pattern can be seen in the CFT domain (see Figure 3.5B), i.e., each tram is made up of different wagons (e.g., traction wagon, regular wagon). Each wagon has a specific function, which is why it is composed of different device groups (e.g. engine controller). These groups are composed by devices (e.g. temperature, location), these are the ones that send information to the monitoring system through attributes, e.g., the device which gives information about the temperature will send the real temperature of the engine in addition to the humidity.

Likewise, in PL domain (see Figure 3.5C), different machines exist, each one with different characteristics. Additionally, although each machine is a compact machine, it has different functionalities that must be monitored (e.g., cluch-break). As in the other domains, these functionalities, which are groups of functionalities, are made up of different devices (e.g. speed sensor) that send data via attributes (e.g., real speed, vibration, time).

A similar structure was identified in the different domains when the information of an ICPS must be collected (see the underside of the Figure 3.5):

"ICPS $\rightarrow$ Logical/Physical Location $\rightarrow$ group of devices $\rightarrow$ concrete device $\rightarrow$ attribute".

Thus, a common data structure can be useful for different ICPSs to be monitored even if they are from different domains.

- **Scalable System:** Considering that different sizes of press machines exist inside a PL, and depending on customers' need the PL it is composed by different machines, it is essential to monitor the ICPS in order the to be aware of what is happening during operation. In addition, the created system needs to be scalable, otherwise the same solution will not be valid for different machine sizes or ICPS sizes. The same occurs with warehouses or even trams. In the case of AW, there are three sizes, their size varies in terms of the number of devices, having from 50 in small warehouses to more than 500 in the case of larger AWs.

  Despite the number of devices, it is important to note that each device can send a large number of attributes. For example, in the case of the tram, for energy control, about twelve devices are installed, but these are capable of sending 400 different attributes. That is why, not all ICPS send the same amount of data to the monitoring system, but all of them must be monitored, hence, the developed architecture must be scalable.

  Thus, the different domains need a scalable solution in order to give response to all ICPS sizes. Different authors also explain the importance of creating scalable systems in the context of ICPSs [WMO$^+$16, ZJP$^+$18].

- **Device Logic Complexity:** Each ICPS is composed by different devices and each has embedded its own logic or function, e.g., in a big warehouse more than 500 devices can be installed and each device is able to send different data depending on the status of the ICPS. For example, in a press machine within a PL, a temperature sensor is located in the clutch-brake.

Imagine that the real temperature exceeds the critical temperature, hence, the temperature is able to send an alert attribute due to that incident. That "new" attribute is never sent before by that device, e.g., if everything is working correctly the real temperature and the timestamp are sent, but in this particular case, the device, besides sending those two attributes, it is able to send the optimal temperature, the limit temperature and a description where the occurred incident is explained. Note that each temperature sensor can have different limits since it is not the same the temperature that a clutch-break can support or the oil temperature of the machine. Thus, considering the number of devices within and ICPS and that each one can have a different logic, managing this variability is necessary because the ICPS becomes uncertain. Even if the provided example is from PL domain, the same situation happens with CFT or AW domain, i.e., the data of the devices within the CFT or AW can change depending on the ICPS status.

- **ICPS Evolution:** The ICPS to be monitored can evolve time to time, i.e., devices can be introduced, deleted or modified during operation. The evolution can be due to many reasons, such as, a device has been damaged and the same device does not exist because it is no longer available, or because of customer needs, it is necessary to replace a device inserting a new one from another provider. There may even be a need to remove, modify or insert devices for business reasons, e.g., it is necessary to start monitoring the temperature of the tram engine which is not being monitored until this moment.

  The internal structure of ICPS itself can also change. For example, an AW grows, i.e., a new floor is added, therefore new devices will certainly be introduced. This makes the structure of the warehouse to be changed. A similar scenario can occurred with CFT, e.g., a new wagon needs to be added since the quantity of wagons was not enough to support all passengers, or in PL, e.g., a new machine is introduced or a group of devices is introduced inside a concrete machine (e.g., press machine) in order to start monitoring another part of the machine which it was not monitored until now. When an ICPS evolves, the evolution can be at any level within its structure, not only at the device level. Thus, all the domains need to manage the evolution of the ICPS to be monitored. Different authors in the literature have recognized ICPS evolution [GPM14, GBH^{+}16, LCK16].

- **Visualization Customization:** In addition to the necessity to monitor an

ICPS, the captured data needs to be converted into information. Thus, it is necessary to visualize the information to the end user in an appropriate way, as this will help him/her in making decisions. Note that in different domains different user roles exist (e.g., in PL domain technical manager, operator; or in CFT the, signalling controller, vehicle controller), that means that not everyone has the same information needs. Thus, the User Interface (UI) needs to be adapted by and for the end user, i.e., an adaptable and adaptive system is needed. An efficient visualization solution must give clear and easy to interpret messages to end users in a timely fashion. Therefore, a good data visualization tailored for individual requirements can provide a better ICPS general overview, faster anomaly detection, which subsequently help users to propose improvements to the ICPS under monitoring.

Additionally, given that an ICPS is highly dynamic (i.e., an ICPS can evolve), an UI must adapted itself based on users' preferences. This need is also identified by Akiki et all. [ABY14a]. In this manner, it would be possible to avoid significant learning and adapting effort required from a user every time an ICPS evolves.

Thus, it is noted that: (I) adapting the UI taking into account the ICPS evolution is necessary, (II) not everybody needs/wants to visualize data in the same way (i.e., different user role exist); (III) teammates collaboration is important since understanding what is happening in the ICPS can be challenging, and (IV) iterating with users for generating new visualization or adapting existent ones is important to help them to work efficiently.

- **Variability on Communication Protocols for Data Transfer:** As mentioned above, devices must send data to a monitoring system. Thus, a communication between the devices and the monitoring system is necessary. In order to achieve that, different Internet of Things (IoT) communication protocols exist, but depending on the industrial domain, the IoT protocols to be used are different. For example, in PL domain, as it is a controlled environment, unless there is a power cut, there are not communication problems in its normalcy. The CFT domain, in contrast, it is a changing environment in which communications may fail more frequently. The movement of the tramway can result in inadequate coverage at certain points of the city, hence, communication problems exist. That is why, depending on the industrial domain to be monitored, it is necessary to analyze its characteristics in order to identify which protocol should be used. It is therefore a variability in terms of the protocol to use across the domains. Additionally, within the same domain, the IoT protocol to use does not have to be the same, this varies depending on the customer requirements.

Thus it can be concluded that: (I) adapting the IoT communication protocol to the customers needs is necessary. (II) In the same ICPS depending on situation the communication protocol can be different. Thus, the communication between the monitoring system and the devices is variable. Different authors used different communication protocols for monitoring their systems [FBT18].

3.3 Revisiting Requirements through a Systematic Literature Review

After this analysis, our challenge is to define a configurable common solution for the management of evolving ICPSs in terms of monitoring and visualization which is: (I) suitable for different domains, (II) scalable for different ICPS sizes and (III) evolution aware with personalized visualizations.

In order to design and develop a common solution, the state of the art through a SLR in terms of monitoring systems is further analyzed. It is evaluated if variability and ICPS evolution when monitoring an ICPS are already recognized and/or addressed in the literature. SLR is used for summarizing existing techniques about a research interest, identifing new research directions, and facilitating the positioning of new research activities [Kee07]. The process followed to carry out the SLR is based on the guideline by [KBB^{+}09]. In this section, a systematic evaluation of the primary studies are reported by a SLR to answer three research questions (RQ):

- RQ1: **What kind of evolution is observed in ICPSs?**
- RQ2: **When, where and how variability is managed in ICPSs?**
- RQ3: **How many primary studies put into practice a system able of managing runtime changes (variability or evolution) in an ICPS?**

In this manner, it is evaluated if variability and ICPS evolution when monitoring an ICPS are already recognized and/or addressed in the literature. If so, can be learned from them, and if not, new solutions can be proposed. With the SLR, it is possible to identify, analyze, synthesize, evaluate, and compare the literature related to the identified requirements. The process followed to carry out the SLR is based on the guideline by [KBB+09].

3.3.1 Process for a Systematic Evaluation of the Primary Studies

The SLR was conducted to answer RQ1, RQ2 and RQ3. To facilitate answering them, seven comparison criteria (CC) were used. First, when the variability is managed in this kind of systems in analyzed. Then, the used techniques to manage variability. Also uncertainty, i.e., if any system considers somehow unknown changes in order to adapt their systems. In addition, what kind of evolution can happen to an ICPS is analyzed. Other analyzed criteria was where this kind of systems are applied, i.e, in which industrial domains. Another criteria about evaluation, was to analyze what kind of research is proposed in the literature in order to solve this kind of challenge. Finally, the maturity of the proposed research is analyzed.

The following CC were used in order to compare different studies:

- **CC1: Binding time** [ASB+09]: When the variability (known or unknown) of the system is (manually or automatically) resolved; (I) Pre-compilation time (while the system is being designed); (II) Runtime (while the system is up and running); and (III) Compilation (while the system is under development and needs to be restarted for applying changes).

- **CC2: Techniques used for variability management:** Which technique is used when variability needs to be managed; (I) Models; (II) Domain Specific Languages (DSLs).

- **CC3: Unexpected changes during operation** [ZSA+16]: At what level the unexpected changes are managed during operation; (I) No; (II) Application Level; (III) Infrastructure Level; (IV) Integration Level.

- **CC4: Evolution in Cyber-Physical System (CPS)** [AWSE16]: In which layer the evolution is managed; (I) Physical Layer; (II) Cyber Layer: Hardware; (III) Cyber Layer: Software

- **CC5: Industrial domain**: In which type of domains this type of systems are found; (I) Manufacturing; (II) Automotive; (III) Medicine; (IV) Others.

- **CC6: Evaluation** [MPFM08]: Which the validation of the proposal is; (I) Validation Research; (II) Evaluation Research; (III) Solution Proposal; (IV) Philosophical Papers; (V) Opinion Papers; (VI) Experience Papers.

- **CC7: Technology Readiness Levels (TRL)** [6]: Which the maturity of the proposed research is; "TRL-1) Basic principal observed; TRL-2) Technology concept formulated; TRL-3) Experimental proof of concept; TRL-4) Technology validated in the lab; TRL-5) Technology validated in a relevant environment (industrially relevant environment in the case of key enabling technologies); TRL-6) Technology demonstrated in a relevant environment (industrially relevant environment in the case of key enabling technologies); TRL-7) System prototype demonstration in an operational environment; TRL-8) System complete and qualified; TRL-9) Actual system is proven in an operational environment (competitive manufacturing in the case of key enabling technologies; or in space)".

Table 3.6: Correspondence between RQs and Comparison criteria

RQs		**Comparison Criteria (CC)**						
		1	**2**	**3**	**4**	**5**	**6**	**7**
RQ1	What kind of evolution is observed in ICPSs?			■	■			
RQ2	When, where and how variability is managed in ICPSs?	■	■			■		
RQ3	How many primary studies put into practice a system able of managing runtime changes (variability or evolution) in an ICPS						■	■

The research question are then linked to the above comparison criteria. By doing so, it allows RQ1, RQ2 and RQ3 to be answered to analyze the other primary studies. The relation between the RQs and the comparison criteria are shown in Table 3.6.

[6] https://ec.europa.eu/research/participants/data/ref/h2020/wp/2014_2015/annexes/h2020-wp1415-annex-g-trl_en.pdf

Based on [Kee07], it is necessary to define the inclusion/exclusion criteria before beginning to analyze the articles. In our case, the following exclusion criteria were used:

- **Non-technical** articles and **non-English** articles were excluded.
- **Excluded testing topic papers**. Once all the papers were downloaded, it was noticed that when reading the titles, many papers were related to testing. Thus, this criterion was defined to exclude papers related to testing, as testing is outside the scope of this dissertation. This comparison criteria was introduced after checking the result of the search string.
- **Not related to our topic**. The articles not related to the topic were excluded (e.g., papers related to variability extraction and System Product Line (SPL) maintenance).
- **Repeated studies**. As the same articles can be reached in different ways, duplicate documents were discarded.

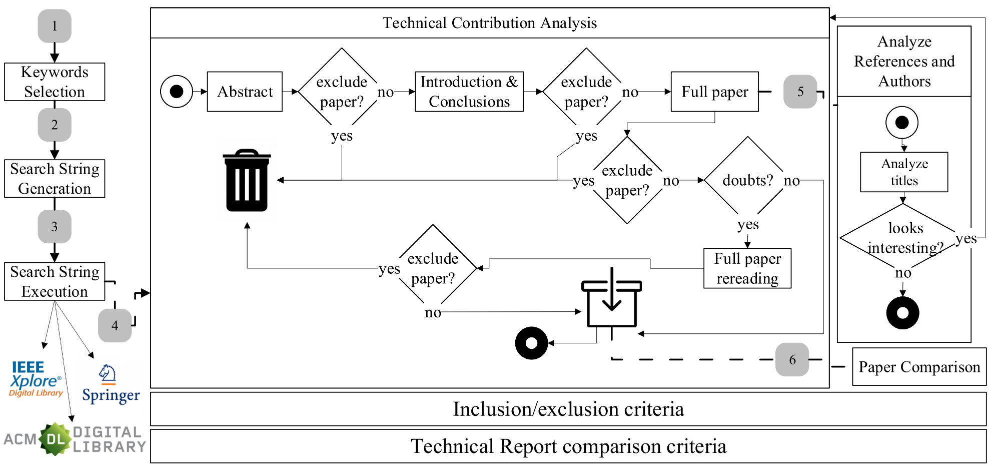


Figure 3.6: The SLR process

Once the comparison criteria and the inclusion/exclusion criteria were established, a systematic process for conducting the SLR was defined, which consists of six steps, as shown in Figure 3.6.

Step 1 - Keywords Selection.

When choosing the keywords to answer questions RQ1, RQ2 and RQ3, focus was placed on the adaptation and evolution of ICPSs. That is why the keywords "adaptation", "adaptive", "evolve", and "evolution" were necessary to find articles related to adaptation in order to handle variability. In the same way, when focusing on ICPSs , the keywords “industrial”, “cyber-physical” and “software” or “system” were necessary. Finally, considering that variability and commonality exist between domains besides within the same domain, "product line" was considered as a keyword.

Step 2 - Search String Generation.

Based on the keywords, the following search string was defined: "industrial" AND "product line" AND ("cyber-physical" or "cyber physical") AND ("adaptation" OR "adaptive") AND ("improvement" OR "improve" OR "evolving" OR "evolve" OR "evolution" OR "refinement" OR "refine") AND ("software" OR "system").

Step 3 - Search String Execution.

Then the search string was manually runed on IEEE Xplore[7], ACM digital library[8], and Springer[9]. These three academic databases were selected because they are research databases with special focus on computer science. According to paperpile in 2018 [10] ACM digital library covers 2.8+ million articles, IEEE Xplore holds 4.7+ million articles and Springer 415000+ articles.

Step 4 - Technical Contribution Analysis.

Based on the results obtained from the databases, the works were analyzed to find relevant contributions to make a comparison of technical contributions considering our research topic. First, the summaries of all articles that were analyzed, excluding articles based on the established criteria. With the articles not excluded and taking into account the inclusion/exclusion criteria, the introduction and conclusions of each

[7] https://ieeexplore.ieee.org/search/advsearch.jsp
[8] https://dl.acm.org/advsearch.cfm
[9] https://link.springer.com/advanced-search
[10] https://paperpile.com/g/research-databases-computer-science/

article were analyzed. Then, the articles considered relevant were read in their entirety, i.e., the ones which were not excluded after analyzing abstract, introduction, and conclusions. The whole process is explained graphically in Figure 3.6.

Step 5 - Analyze References and Authors.

Once all articles were analyzed, the authors and references in the area (see Figure 3.6 (right)) were analyzed . The selected authors and references were chosen from the papers that were thoroughly read. To select them, the title was read, and if it was considered suitable for analysis, it was analyzed. The selected ones were analyzed using the "Technical Contribution Analysis" process, i.e., Step 4.

Step 6 - Paper Comparison.

Finally, all papers were compared by the comparison criteria established in a systematic way making it possible to reach an objective conclusion.

3.3.2 Results of the Systematic Literature Review

As shown in Table 3.7, for each established comparison criterion, each paper was checked to label it as: (I) no information provided on the criterion (not information), (II) partially meeting the criterion (if it is explicitly mentioned in the article but not fully addressed or if it is insufficiently clear if the criterion is met (partially achieved)), and (III) fully meeting the comparison criterion (achieved).

After the keyword selection and executing the string search, 123 papers were obtained from three different databases (Springer: 67; ACM: 12; IEEE: 45). Considering the exclusion criteria, 17 papers were outside the scope, i.e., in the testing area, one of which was not written in English and six of which were not technical articles. Thus, 24 papers were excluded and eventually, 99 papers were obtained (Figure 3.7 (Left)).

Once the abstracts of the 99 articles were read, 54 papers were excluded. However, 15 out of 45 (selected articles) were still doubtful. The introductions and conclusions were then read, from which other 24 papers were excluded, descending them to 21 papers for reading the full text. After the first full-text reading, six papers were excluded, and 5 were doubtful. Therefore, the five papers were read again and four

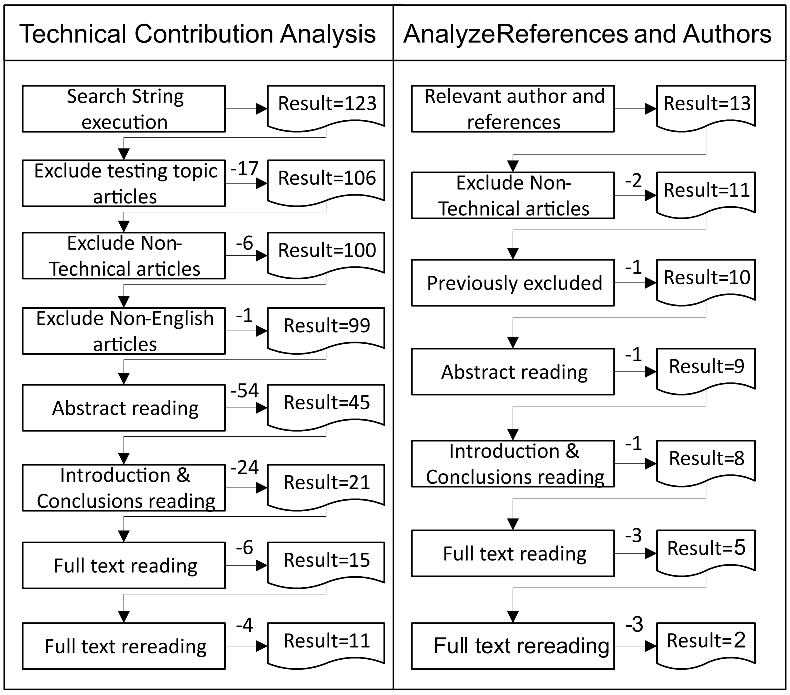


Figure 3.7: Left) excluded papers in technical contribution analysis (Step 4). Right) excluded papers in authors and reference analysis (Step 5)

of them were excluded. Thus after the SLR process, 11 papers were systematically selected to be compared. The summary of the discarded papers is shown in Figure 3.7 (Left).

Subsequently, in order not to exclude any interesting paper, all the authors and references of those papers that were completely read were analyzed. Because these authors may have related articles which can be interesting or have reference to interesting articles. Thus, a total of 13 articles where analyzed. As shown in Figure 3.7 (right), a total of 11 papers were excluded from the 13 articles. Thus, two papers were selected to be compared. To do so, the same process followed with technical contribution analysis was followed (Step 4). Finally, as shown in Table 3.7 all the selected papers (a total of 13) were compared by the article comparison criteria established above.

Table 3.7: SLR results

□ No information provided ▲ Partially achieved ■ Achieved		[TLWD18]	[RSY+17]	[SPO+17]	[ASFV18]	[MRWH15]	[CZH17]	[BMC+17]	[BQ15]	[QRV+15]	[HR16]	[MNSY18]	[LBK15]	%
CC1: Binding time	Pre-compilation		■	■	■	■	■							41.6%
	Runtime	■						■	■	■	■	■	■	58.33%
	Compilation		■		■	▲								20.83%
CC2: Variability management techniques	Models	■	■	■	■	■	■	■	■	■	■	■	■	100%
	Domain Specific Languages			■								■		16.6%
CC3: Uncertainty during operation	No	□	■	■	■	■	■	■			□	■		70%
	Application Level								■	■			□	18.18%
	Infrastructure level									□				0%

Continued on next page

Table 3.7 – *Continued from previous page*

☐ No information provided ▲ Partially achieved ■ Achieved			[TLWD18]	[RSY+17]	[SPO+17]	[ASFV18]	[MRWH15]	[CZH17]	[BMC+17]	[BQ15]	[QRV+15]	[HR16]	[MNSY18]	[LBK15]	%
	Integration Level										☐				0%
CC4: Evolution in CPS	Physical Layer							■							8.33%
	Cyber Layer	Hardware						■				■			16.67%
		Software	■	■	■	■	■	■	■	■	■	■	■	■	100%
CC6: Industrial Domain	Manufacturing		■					■	■						25%
	Automotive			■		■					■		■		33.33%
	Others				■					■		■		■	33.33%
	Medicine						■								8.33%
CC7: Evaluation	Validation Research		■									■			8.33%
	Evaluation Research					■			▲						12.5%

Continued on next page

Table 3.7 – *Continued from previous page*

☐ No information provided ▲ Partially achieved ■ Achieved		[TLWD18]	[RSY+17]	[SPO+17]	[ASFV18]	[MRWH15]	[CZH17]	[BMC+17]	[BQ15]	[QRV+15]	[HR16]	[MNSY18]	[LBK15]	%
	Solution Proposal		■	■		■	■		■	■		■	■	66.67%
	Philosophical Papers													0%
	Opinion Papers													0%
	Experience Papers													0%
CC8: Level of assessment TRL	TRL-1							■					■	16.67%
	TRL-2					■								8.33%
	TRL-3	■		■			■	■		▲		■		54.17%
	TRL-4										■			8.33%
	TRL-5		■		■									16.67%
	TRL-6													0%

Continued on next page

Table 3.7 – *Continued from previous page*

☐ No information provided ▲ Partially achieved ■ Achieved		[TLWD18]	[RSY+17]	[SPO+17]	[ASFV18]	[MRWH15]	[CZH17]	[BMC+17]	[BQ15]	[QRV+15]	[HR16]	[MNSY18]	[LBK15]	%
	TRL-7													0%
	TRL-8													0%
	TRL-9													0%

After comparing all the primary studies in the specific research topic of this dissertation, the research questions (RQs) stated at the beginning of the section, i.e., RQ1, RQ2 and RQ3, can be answered:

- **RQ1: What kind of evolution is observed in ICPSs?** It can be concluded that all the authors' consider software evolution, but few considered hardware evolution (16.67%). Additionally, very few contributions consider managing unexpected changes, i.e., 18.18% of the authors consider unexpected changes in application layer.
- **RQ2: When where and how variability is managed in ICPSs?** Taking into account the obtained results, the 41,67% manage changes variability at design time (pre-compilation), the 58,33% at runtime and in the 20.83% of the cases the variability is managed when the system compiled. The most popular domains where ICPS variability management is analyzed are automotive and manufacture domains (58.33%). Additionally, as observed in Table 3.7, all the solutions rely on models for managing variability.
- **RQ3: How many primary studies put into practice a system able of managing runtime changes (variability and evolution) in an ICPS?** The 45.83% of the contributions have an experimental evaluation (TRL-3), i.e., not tested in a real industrial environment. Moreover, very few papers were demonstrated in a relevant environment, i.e., about 16.67%, since most of the papers are solution proposals (66.67%). Also, those who consider unexpected changes do not have an evaluation associated with the given solution.

Considering the analyzed contributions, it can be concluded that even if many primary studies considered ICPS evolution and also variability, very few consider unexpected changes, i.e., they do not consider not designed changes. In addition, the primary study that considers unexpected changes in the physical and cyber layers does not consider runtime evolution [CZH17]. Moreover, all the primary studies used models for managing variability in ICPSs.

Thus, it is noted that managing evolution inside an ICPSs is more and more common but, considering the analyzed papers, none of them recognize it in the literature. Since mostly the evolution is managed in software instead of in hardware, that means that the evolution of introducing, deleting or updating devices inside an ICPS is not sufficiently recognized in the literature.

PART III

CONTRIBUTION

"If you're going to be able to look back on something and laugh about it, you might as well laugh about it now."

MARIE OSMOND

Chapter 4

Modular Monitoring System

This Chapter[1] presents a *modular monitoring and adaptable & adaptive visualization solution for facing the evolution of Industrial Cyber-Physical Systems*. This solution is composed by two main systems: the *modular monitoring system* and the *personalized visualization & evolution detection system*.

More precisely in this chapter, the *modular monitoring system* is detailed. With such system, it is possible to configure different Industrial Cyber-Physical Systems (ICPSs), start monitoring them and make cyberphysical changes (e.g., launch and start monitoring a new device) at runtime without stopping the monitoring system.

The content of the Chapter is structured as follows: first, the general solution is presented. Then, the *modular monitoring system* is described followed by the main components of the *modular monitoring system*. Next, the operation of the *modular monitoring system* is explained. Finally, the Chapter is concluded.

Contents

4.1	Introduction	**62**
4.2	Modular Monitoring System	**66**
4.3	Modular Monitoring System Components	**68**
4.3.1	ICPS Composition based on ISA-95	68
4.3.2	Agent-Based Communication	69
4.3.3	Data Representation by IEC 61850 and IEC 62264	72
4.4	Modular Monitoring System Configuration	**75**
4.4.1	Agent Generation	75
4.4.2	Monitoring System Configuration	77
4.5	Conclusion	**80**

[1]The content of this Chapter has partially been published in [ILA$^+$17, ISA19, IIL$^+$19b]

4.1 Introduction

After conducting the domain analysis some commonalities were identified in order to supervise an ICPS: (I) a modular monitoring system that needs to be (II) scalable and (III) valid for different ICPS structures, besides (IV) being able to manage device data complexity. In addition, an adaptable and adaptive visualization system able to (V) manage ICPS evolution and (VI) be able to customize the User Interface (UI). To address those requirements, a *modular monitoring and adaptable & adaptive visualization solution for facing the evolution of Industrial Cyber-Physical Systems* is proposed.

In order to meet the identified requirements the next items need to be fulfilled by our solution:

- **Monitoring System:** A multi-domain system which captures data from an ICPS at runtime.
- **Communication Protocols:** Support for different communication protocols.
- **Structured Data:** Support for ICPS evolution caused by the user or by the device.
- **Evolution Detection:** Detect ICPS evolution and classify it.
- **Adaptable Visualization:** Customize the visualization by the user.
- **Adaptive Visualization:** Adapt the visualization to the end user considering the ICPS evolution.

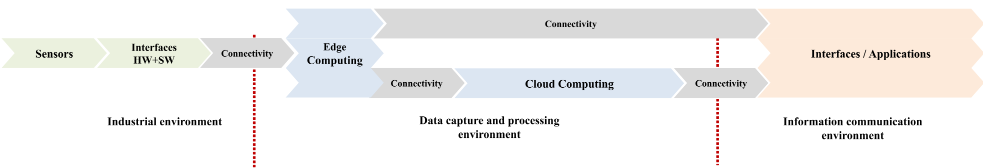


Figure 4.1: All data stages from its capture to its visualization in order to supervise ICPSs.

In order to supervise an ICPS, it is necessary to cover all data stages as shown in Figure 4.1. The *industrial environment*, (e.g., sensors, actuators, displays) is linked with the *data capture and processing environment* (e.g., Internet, Cloud Computing, Edge Computing) in order to transfer the information to end user, i.e., *information communication environment*. Below each environment is explained:

- **Industrial Environment:** This is the physical location of all devices. These devices will be monitored and controlled in order to collect data from them.
  - **Devices:** It refers to physical elements (hardware) that provides useful functionality and/or information when there are connected to a computer [LR02], i.e., it is a hardware which is integrated in the production equipment [IEC13b], e.g., temperature sensor.
  - **HW + SW interfaces:** It refers to the software included in the device in order to receive or send data.
  - **Connectivity:** It refers to the communication between the *Industrial environment* and *Data captured and processing environment*.
- **Data Capture and Processing Environment:** The captured data from an ICPS is analyzed and/or manipulated in order to be stored, apply machine learning techniques, make data analytic, etc.
  - **Edge Computing:** It refers to technologies that enable the calculation to be carried out at the edge of the network, on downstream data on behalf of Cloud Services and on upstream data on behalf of Internet of Things (IoT) Services [SCZ+16]. This is where data acquisition and initial processing occurs.
  - **Cloud Computing:** The central data center solution where complex overview analytics are performed, high availability data persist, and the environment is established to provide advanced data services. In this case, it is necessary to consider aspects such as elastic resource management, scalability, and availability of resources.
  - **Connectivity:** Two different connectivities exist: (I) between two regions within the same environment (e.g. Edge and Cloud Computing) or (II) with other environments, in this case with *Information Communication Environment*.
- **Information Communication Environment:** In this environment, the data is converted into information and it is transferred to the end user in order to make decisions. It is the environment where the user can interact somehow with the captured data as well as with the monitored ICPS.

- **Interfaces/Applications:** This region is where the information is transferred to the end users, usually powered by an interface, i.e., Human Machine Interaction (HMI).

Bear in mind that both *Edge Computing* region and the *Cloud Computing* region are capable of transferring data to the *Information Communication Environment* in order to visualize captured data from an ICPS. Depending on the customer requirements the *Cloud Computing* region is optional, as not all ICPSs need to transfer data to the cloud since the calculations may be conducted locally.

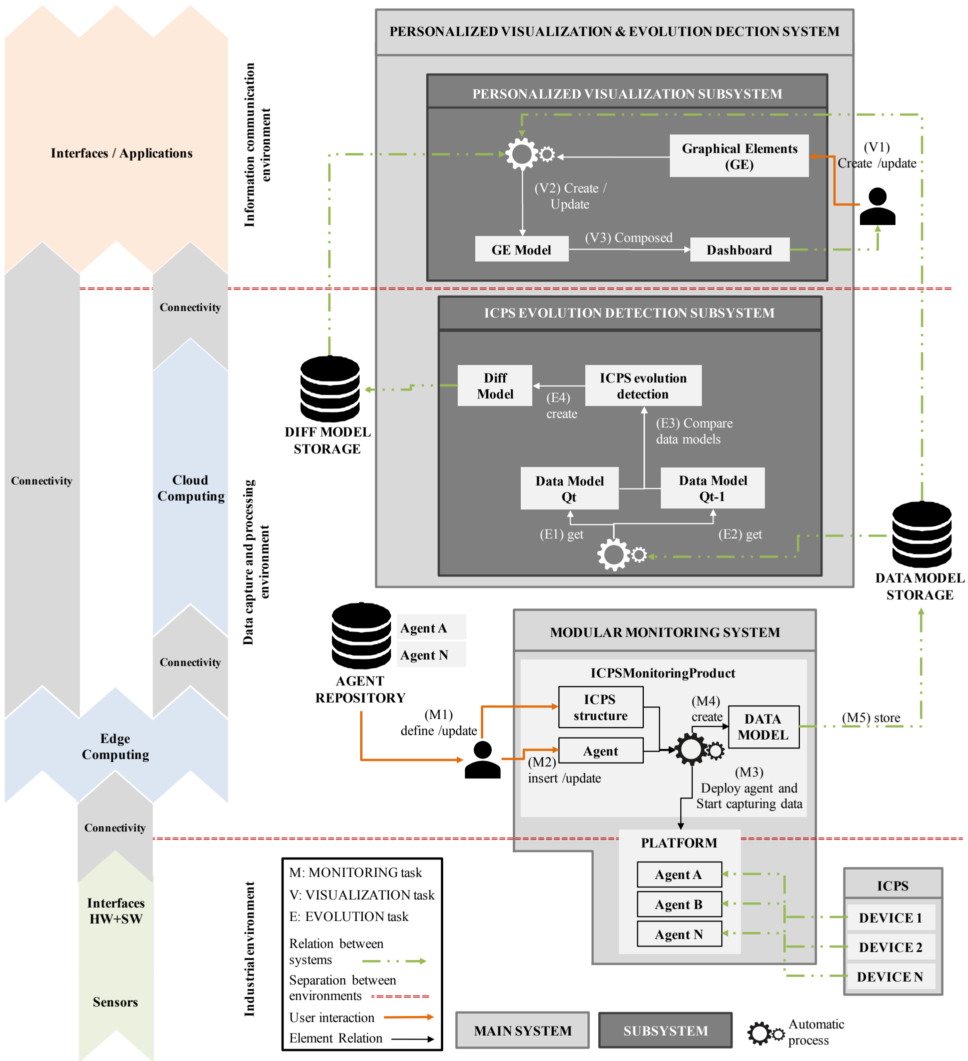


Figure 4.2: Solution Overview

In order to cover all data stages and the presented requirements, a *modular mon-*

itoring and adaptable & adaptive visualization solution for facing the evolution of Industrial Cyber-Physical Systems is proposed. This solution is based in two main systems as shown in Figure 4.2: (I) a ***modular monitoring system*** for evolving ICPSs; and (II) the ***personalized visualization & evolution detection system***.

The *modular monitoring system* is able to: (I) capture data from the devices in the ICPS and then (II) store them in a structured way. As every ICPS is different, the *modular monitoring system* allows to configure a concrete monitoring system that monitors a concrete ICPS, i.e., a *ICPSMonintoringProduct*. Every *ICPSMonintoringProduct* is a distributed system based on *agents*. An *agent*, which is deployed automatically in the *Platform* (i.e, the software that manages the life-cycle of an agent) by the *ICPSMonintoringProduct*, is a software responsible of capturing data from the *devices* and transferring it to the *ICPSMonintoringProduct* using a communication protocol. One *agent* per device is necessary and the *platform* is a PC/Machine responsible of launching each *agent*. In order to configure a *ICPSMonintoringProduct*, the user needs to: (I) configure the ICPS structure, and (II) introduce/update the agents. Every industrial domain contain an agent repository where different agents are stored in order to reuse them between plants.

After the *ICPSMonitoringProduct* is configured, this is responsible of deploing every *agent* in the corresponding *platform* and start capturing the data from the devices every timestamp. Then the *ICPSMonitoringProduct* organizes captured data in a structured way using models, i.e., once the data is received, a *Data Model* is created. The *Data Model* conforms a *Data MetaModel*. The *Data MetaModel* is based on two international standards, International Electrotechnical Commission (IEC) 62264 and IEC 61850. Every *Data Model* reflects the physical/logical organization of the ICPS in a certain point in time. From the combination of *agents* and the *Data MetaModel*, is possible to model the ICPS every timestamp even if an evolution has taken place. The data processing is usually made in the *Edge Computing* region, since it is the closest region between the devices and the *Data capture and processing environment*. Every timestamp, the Data Model is stored.

In order to display the data stored by the *ICPSMonintoringProduct* (*Data Model*s), the *personalized visualization & evolution detection system* is used. This system is capable of visualizing the data in a personalized way to the end user in addition to detecting and effectively classifying ICPSs evolution at runtime and adapting the visualization. To do so, the *personalized visualization & evolution detection system* is divided in two subsystems: (I) *ICPS evolution detection subsystem* and (II) *personalized visualization subsystem*. The *ICPS evolution detection subsystem* is able to detect and classify the evolution (Diff Model). In the same manner, the *personalized visualization subsystem* is able to: (I) create Graphical Elements (GE) at runtime by interacting with users and considering data captured from *ICPSMonintoringProduct* (Data Models); (II) update the UI by considering ICPS evolution and inform users about the changes; and (III) allow users to create new visualizations or update existing ones at runtime, and share visualizations with others.

In this Chapter, the *modular monitoring system* is explained in detail, whereas in the following chapter, the *personalized visualization & evolution detection system* is presented in detail.

The rest of the Chapter is structured as follow. First, the *modular monitoring system* is introduced. Then, the composition of the system is described followed by how the systems works. Finally, the Chapter is concluded.

4.2 Modular Monitoring System

The *modular monitoring system* is a software system that allows the user to configure an ICPS (*ICPSMonitoringProduct*) in order to collect and map the captured data to finally store that data in a structured way.

The *modular monitoring system* is based on the combination of different international standards. The main challenge of the *modular monitoring system* is that: (I) it is able to monitor different ICPSs, with different ICPS structure or composition, in addition to the fact that the ICPS to monitor can belong to different industrial domains and (II) it is able to handle ICPSs evolution at runtime.

The conceptual model of the *modular monitoring system* is represented in the Figure 4.3 by a Unified Modeling Language (UML) class diagram. As shown in the Figure, the *modular monitoring system* can represent different ICPS domains, i.e., it is possible to configure different monitoring systems (*ICPSMonitoringProduct*s) valid for different domains (e.g., Press Line (PL), Automated Warehouse (AW)).

In order to give a personalized solution, the *modular monitoring system* contains the following components: (I) ISA-95 based ICPS structure; (II) Agent-Based Communication in order to communicate the ICPS with the *ICPSMonitoringProduct*; and (III) IEC 61850 & IEC 62264 based Data Representation (Data Model) where all the

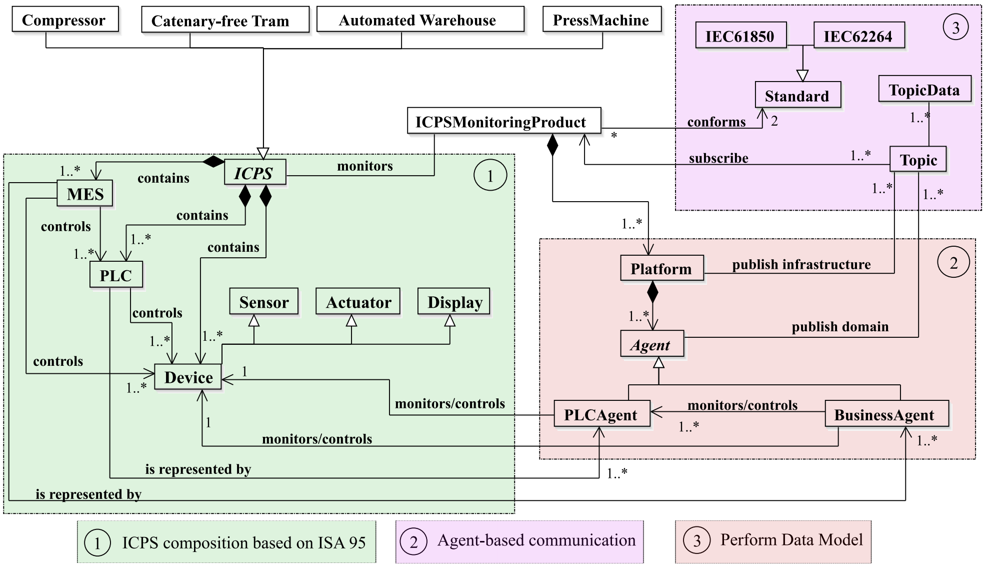


Figure 4.3: Conceptual Model of the *Modular Monitoring System*

captured data is stored in an structured way.

- **ISA-95 based ICPS structure:** Every ICPS to be monitored is structured with ISA 95 [ANS05], a well-known and widely used international standard for developing and integrating ICPSs. This standard helps to understand communications between devices and monitoring system.

- **Agent-Based Communication:** An ICPS is composed by different devices which are physically distributed and each one is able to send different data. Thus, in order to manage each device independently, agent-based communication is used. Every agent encapsulates functions of monitoring & controlling a concrete device, which enables easier managing of changes [KVA13]. In this way, each device has its own communication protocol and logic.

- **IEC 61850 & IEC 62264 based Data Representation:** Data MetaModel allows to capture all the data from devices in a structured way. In addition, it is possible to have the traceability of each device and the whole ICPS. The Data MetaModel is based in two standards: IEC 62264 and IEC 61850.

4.3 Modular Monitoring System Components

In this section the main parts of the *modular monitoring system* are presented in detail. In Section 4.3.1 the ICPS composition based on ISA 95 is discussed. In Section 4.3.2, the Agent-Based Communication between an ICPS and its monitoring software system (*ICPSMonitoringProduct*) and in Section 4.3.3, how received data from the devices is stored in a Data Model is explained.

4.3.1 ICPS Composition based on ISA-95

The IEC 62264 (IEC 62264-1:2013) [IEC13b] and its predecessor ISA-95 standard [ANS05] are the widely-used international standards for developing and integrating ICPSs in domains such as Computer Numerical Controls (CNCs), compressors, Catenary-Free Tram (CFT), and AW [PBM07]. The ISA-95 allows to organize all the different components (devices) within an ICPS. Thus, ICPSs from different domains can be represented using this standard.

Even if the ISA-95 standard is composed by four different levels (see Section 2.1 in Chapter 2), this dissertation is currently focused up to Level 3 because supervising and controlling industrial domains, production lines, and devices are the critical requirements for different industrial domains.

Every ICPS is composed by different *devices*, which can be classified into the following three categories:

- *Sensor*: It captures data about a device or its process.
- *Actuator*: It receives and executes orders to change certain aspects of a device or its process.
- *Display*: It receives and visualizes data.

In order to control the devices, these ones are classified in two main groups based on ISA-95, i.e., *Manufacturing Execution Systems (MESs)* and *Programmable Logic Controllers (PLCs)*. Both of them can control *devices* directly in an *ICPS*. A *MES* can also indirectly control *devices* via a *PLC* (see Figure 4.4).

4.3.2 Agent-Based Communication

An Agent is an autonomous software able to make decisions in a concrete environment [Woo97]. An agent has different properties, i.e., it is autonomous, reactive, pro-active and a system with social abilities (able to interact with other agents) [WC00]. In this Ph.D. dissertation an agent is refered as an autonomous software with monitoring and controlling functionalities that is able to monitor or control a device.

The agents allow to handle each device separately. Additionally, an agent has communication capabilities, i.e., it is the link between the physical device and the monitoring system. Each agent controls or monitors a device using a specific communication protocol, i.e., one agent per device needs to be developed. In this manner, every agent will provide flexibility and adaptation capability to the solution, not only for software, but also for the inclusion, removal, and modification of physical devices since all the devices are decoupled from each other.

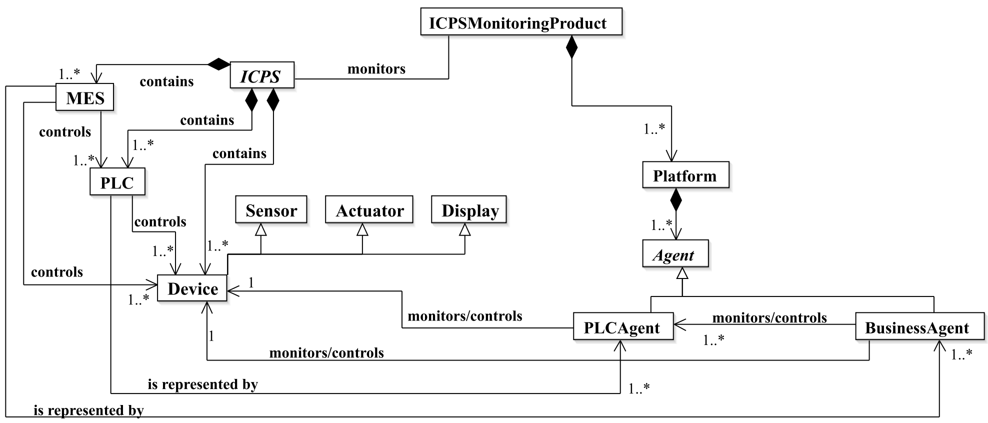


Figure 4.4: Agent-Based Communication between ICPS and monitoring software system

In order to start capturing data through agents, these ones need to be launched. To do so, a *platform* is necessary. A *platform* is a PC/machine with a specific software that is responsible of launching or stopping the agents. More than one *platform* can exist in every ICPS (see Figure 4.4). This depends on the quantity of agents and the capacity of each PC/machine. The *platforms* allow dividing agents into different PC/machines, as deploying all agents to the same machine would be unfeasible, i.e., not enough memory/CPU on a unique PC.

Agent Types

To monitor *ICPSs* at different levels, two types of *agents* are defined based on the ISA-95 standard (see Figure 4.4):

- *BusinessAgent*: Representing supervision at the *MES* level (level 3) and monitoring & controlling an *ICPS* with the capability of making decisions.
- *PLCAgent*: Representing monitoring & controlling of *devices* at the *PLC* level (level 2).

BusinessAgents are intelligent (able to make decisions) software that can receive data directly from *devices* and dispatch them actions based on decisions made. A *BusinessAgent* can control a set of *PLCAgents*, and if the expected data is not received, it can make decisions, e.g., communicating an error in a *PLCAgent* to the monitoring system (i.e., ICPSMonitoringProduct). *PLCAgents* receive direct data from devices, but they do not make decisions by themselves; they are only able to dispatch actions that have received from *BusinessAgents* or get data from devices.

Agents Characteristics

As shown in Figure 4.5, the structure of an agent is defined in order to capture data from the a device and transfer to the monitoring system:

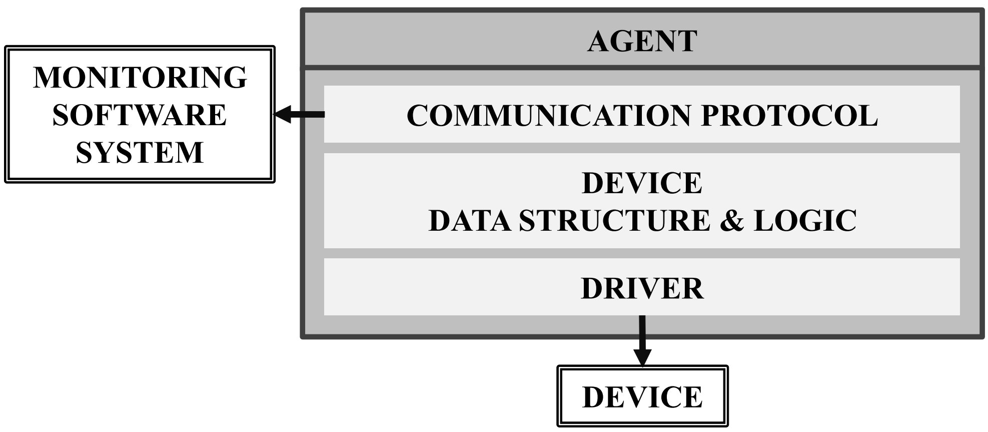


Figure 4.5: Agent Composition

- **Communication protocol:** Different communication protocols can be used in order to transfer data to the monitoring system. Some of the most used communication protocols are Data Distribution Service (DDS), Simple Object Access Protocol (SOAP), Hypertext Transfer Protocol Secure (HTTPS) and Constrained Application Protocol (CoAP) as mentioned in Chapter 2.
- **Device data structure & logic:** An agent needs to be developed based on the IEC 61850 standard [IEC13a]. Although the standard is for monitoring and controling of electrical substations, it is shown to be also usable outside this domain [PHA$^+$10]. The IEC 61850 standard is a standard that defines a Basic Information Model, services to interact with the device, and some recommendations for the use of different communication protocols. In addition, it also has Control Blocks (CB) for additional functions (e.g. the transmission of different attributes depending on the state of the system). More information about the IEC 61850 in Appendix B.
- **Driver:** A program that controls at low level a particular type of device that is attached to a system.

In order to facilitate the creation of agents, a tool is designed and developed in collaboration with another Ph.D. student [2]. The designed and developed tool is a sof**T**ware p**R**oduct l**I**ne based mu**L**tidomain iot **A**r**T**ifact g**E**neration for indust**R**i**AL** cps (TRILATERAL).

TRILATERAL is a Model-Based Engineering (MBE) tool that uses the System Product Line (SPL) paradigm and Domain Specific Language (DSL) to make it easier for the user to graphically configure the IoT *communication protocols* in order to monitor/control the ICPS and the *device data structure & logic*. Therefore, with TRILATERAL, an IoT communication middleware is created.

TRILATERAL is divided into two parts: (I) the *Server Model* definition and (II) the *Information Model* definition. Both of them are configured by the user using the DSL. The former one, i.e., *server model* definition, is used for configuring the IoT communication protocols for data transition (e.g., when the data from the device needs to be transferred to the monitoring system). Currently, TRILATERAL provides three IoT communication protocols, i.e., WS-SOAP, HTTP-REST and CoAP. The latter one (i.e., *Information Model*), is for configuring the *device data structure & logic* based on IEC 61850.

[2]http://orcid.org/0000-0001-7708-3252

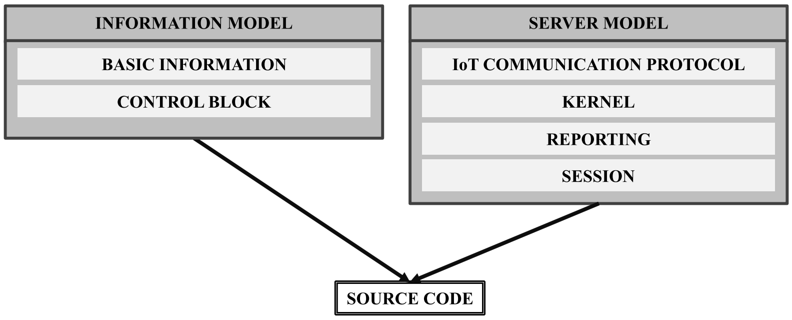


Figure 4.6: TRILATERAL components

In order to configure the *Information Model*, the IEC 61850 was modeled. Thanks to it and the DSL, the intelligence/logic of the device can be configured. To do so, the *Basic Information Model* and *Control Block* of the IEC 61850 need to be configured (see Figure 4.6). The *Basic Information Model* defines the elements of the real world (Logical Device, Logical Node, Data, DataAttribute), i.e., the structure of the device within the ICPS to be monitored. The *Control Blocks* are specialized classes to interact with the information model through some additional functionalities. More information about the mapping and implementation in [IUUPCM17, IUCMMU18]

In the same manner, to configure the *Server Model*, the user needs to select and configure the communication protocol to use, the kernel, the reporting and also the session by using the DSL. Thanks to the DSL and the user collaboration, is possible to automatically configure the server in order to communicate the *agent* with the monitoring system, i.e., *ICPSMonitoringProduct*.

4.3.3 Data Representation by IEC 61850 and IEC 62264

In order to capture data in a structured way, two standards are analyzed. The IEC 61850 defines guidelines for monitoring and controlling devices within an ICPS. IEC 62264 (its predecessor ISA-95) standard describes interfaces between manufacturing control and enterprise functions and it is part of RAMI 4.0 [HB15], which provides monitoring of different industrial environments of the same industrial domain.

Taking into account that IEC 62264 is based on ISA-95 and the agent is configured based on IEC 61850 as mentioned in previous section, the combination of both is proposed in order to structure all the data coming from the ICPS that is being monitored.

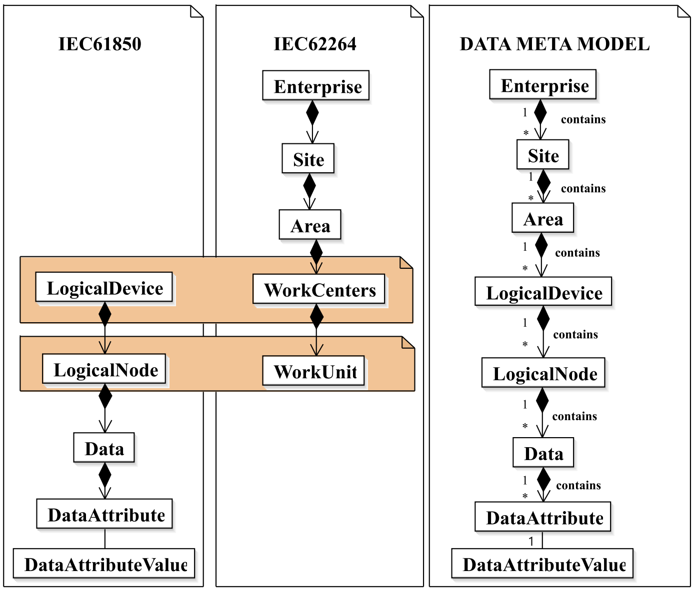


Figure 4.7: Data Model Structure based on the IEC 61850 and IEC 62264 Standards

Table 4.1: Data MetaModel level description

Data MetaModel Level	**Descriptions**
Enterprise	ICPS identification, company name.
Site	Geographical or physical distribution of the ICPS.
Area	Logical distribution inside the ICPS.
Logical Device	Devices description.
Logical Node	Device identification.
Data	Information description that the devices send.
Data Attribute	Concrete information that the device sends inside the data.
Data Attribute Value	Concrete value that the device sends in each timestamp. It is the concrete value of the Data Attribute.

Additionally, after studying guidelines from the two standards, i.e., the purpose of the standard, what it should be used for, etc., similar concepts with the same functionality were identified, which are shown in Figure 4.7. It was observed that, the *Logical Device* and the *Work Center* represent the equipment or the physical device, instead, the *Logical Node* and the *Work Unit* describe the *Logical Device* or the *Work Center*. Additionally, the *Work Unit* is the lowest level of the hierarchy and the *Logical*

Device is the highest level, as long as you do not want an agent to control more than one device. Thus, as shown in the right-side part of Figure 4.7, the structure of the *Data MetaModel* was defined by integrating these two standards, which characterizes the data to be monitored in an *ICPS*.

In the Table 4.1 all the levels of the *Data MetaModel* are described. The Enterprise, Site and Area are the organizational structurs of the ICPS to monitor. Every device within the organizational structure is represented by the combination of *Logical Device* and *LogicalNode*, and each *Logical Device* can have many topics. Each topic represents *data* that a specific *device* can send; when a *topic* takes a particular value, which is considered as a *TopicData*.

Thus, the Data MetaModel is the artifact that allows representing both the data and structure at once it. Contains seven levels that represent the logical and physical structures of different ICPSs (see Table 4.1). In order to be valid this Data MetaModel in different ICPSs, the following requirements need to be considered:

- **Quantity of levels:** The Data Model conformed by the Data MetaModel will always be composed by 7 levels, i.e., there cannot be a branch containing only 5 of them. This is to ensure complete traceability of the system at every moment, i.e., have the relationship between all devices and the organizational structure of the ICPS.
- **ICPS representation:** The Data MetaModel has a hierarchical structure since it is shown in the literature that a tree model structure facilitates the detection of an evolution [SS08]. This implies that a node of the Data Model cannot depend on several nodes, i.e., a single node contains a single parent node.

Thus, when an ICPS is monitored a Data Model is defined by the *ICPSMonitoringProduct* every timestamp. Notice the ICPS to monitor can evolve making every Data Model different. Table 4.2 provides an example of the kind of data that is stored in each element of the Data Model.

When a device starts sending data via an agent, the monitoring system is able to manage received data. The monitoring system joins the data received from the agent, i.e, device data structure that it is based on IEC 61850, with the ICPS structure defined by the user, i.e., IEC 62264. Thus, a concrete *Data Model* reflects an snapshot of an *ICPS* at a particular moment in time which is an instance of the *Data MetaModel*.

Table 4.2: An Example Data Model

Data MetaModel Level	**AW example**	**PL example**	**CFT example**
Enterprise	MyWarehouse	Company	Tram
Site	Mexico	Madrid	Olso
Area	Group6	LeftZone	Regular Wagon
LogicalDevice	Scanners	Machine4	Batery Controller
LogicalNode	Scanner8	ClutchBreak	Batery
Data	State	Temperature	Capacity
DataAttribute	Enable	Highest	Percentage
DataAttributeValue	0	50	20

4.4 Modular Monitoring System Configuration

In this section the workflow of the configuration of the *modular monitoring system* is presented. As a first step, how an agent using TRILATERAL and user collaboration can be created is explained. As a second point how concrete monitoring system can be configured by the user and how it works is explained.

4.4.1 Agent Generation

As show in Figure 4.8, first, the user configures the *server model* by choosing the IoT communication protocol (step 1). More than one IoT communication protocol can be configured if required. Hence, the user must choose the protocol that best suits their ICPS. Figure 4.9 shows the feature model where the combinations that the user can choose to configure the *server model*.

Once the IoT communication protocol is chosen and the user has selected all the configurations it needs, TRILATERAL automatically creates user and server configuration files. Then, two directories are automatically generated: one for certificates, where all security certificates are stored; and another one where the files related to the file management functionalities offered by the IEC 61850 standard are stored.

After the communication protocol is chosen, the user configures the *information model* based on IEC 61850 (step 2). In Appendix B in Figure B.1 the detailed

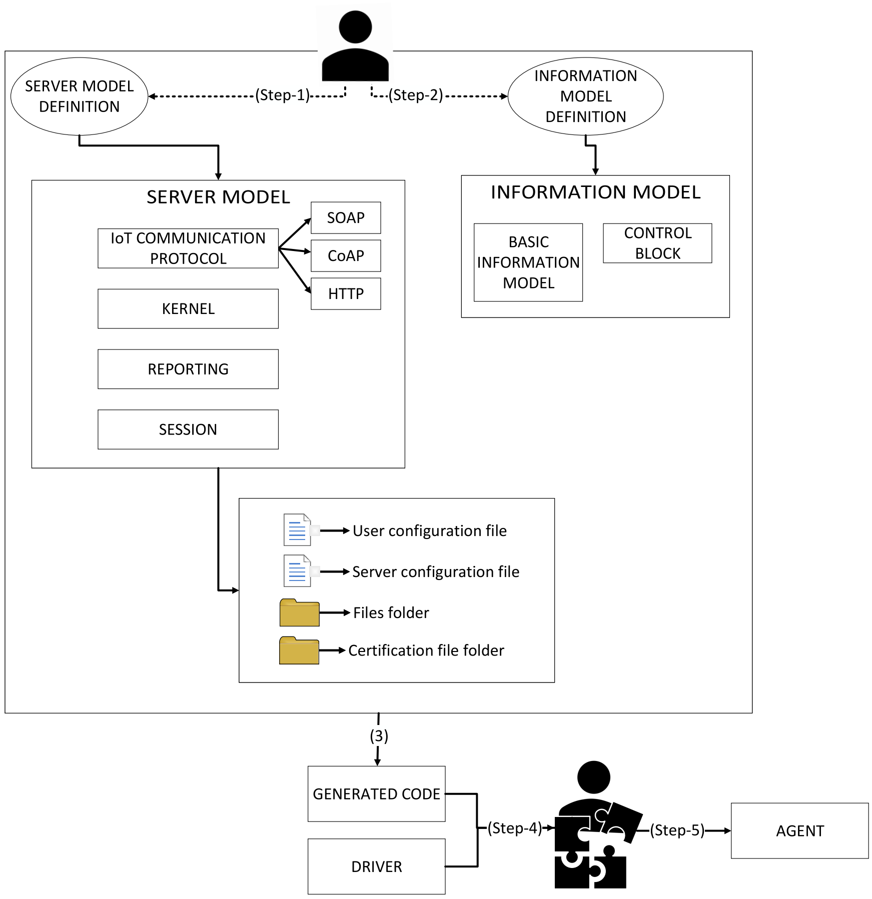


Figure 4.8: TRILATERAL workflow

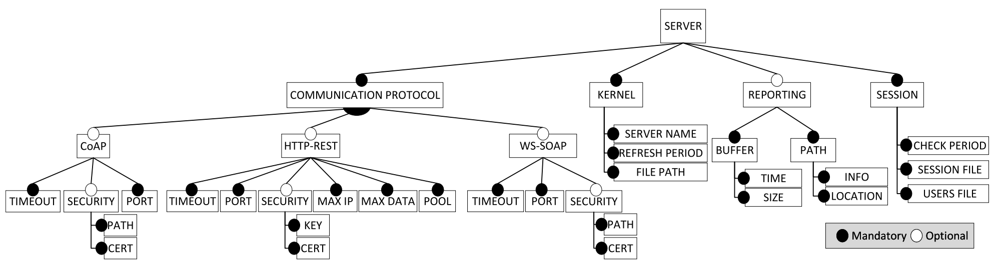


Figure 4.9: Server definition feature model

information model of the IEC 61850 is presented. In order to configure the ICPS based on IEC 61850, the user configures the *Basic Information Model* and the *Control Blocks*. Then TRILATERAL automatically generates the code which corresponds to the *communication protocol* and the *device data structure & logic* (*generated code*). To do so, model to text (M2T) transformation is used (step 3).

In order to create the agent, the user needs to develop the corresponding driver (the connector that links the devices with the *information model*) and compile the *generated code* with a specific driver (step 4). Once the driver and the device logic are compiled with the *generated code*, some libraries and executables are created (step 5), i.e., *the agent*.

Once all the agents are created, these ones need to be introduced in the modular monitoring system in order to configure the *ICPSMonitoringProduct*. Then, the *ICPSMonitoringProduct* is responsible to deploy the agents in the corresponding platform in order to start capturing data from the devices. The relation between TRILATERAL and the *ICPSMonitoringProduct* is shown in Figure 4.10.

4.4.2 Monitoring System Configuration

As shown in Figure 4.10 the *modular monitoring system* contains at least one platform and a *ICPSMonitoringProduct*. In the platform the agents that monitor the devices are launched. In order to join the platforms with the *ICPSMonitoringProduct* it is necessary to deploy specific software in each PC/machine manually. This software is generic and it is responsible of receiving agents and making them work, i.e, it is responsible for launching them. Taking into account that multiple platforms can exist, it is necessary to identify them, hence, each platform will have a specific configuration file where the platform identification will be specified.

Once the specific software is launched in each platform, it is necessary to configure the concrete *ICPSMonitoringProduct* in order to start monitoring an ICPS. One *ICPSMonitoringProduct* for each ICPS. As shown in Figure 4.11 first, every platform needs to be register in the *ICPSMonitoringProduct* (*1: configurePlatformConfiguration()*).

Then, the *organizational structure* and the *agents* need to be defined. In the *organizational structure* the logical/physical distribution of the ICPS to be monitored needs to be specified (*3: insertOrganizationalStructure(Enterprise, Site, Area)*), specifically, the part that corresponds to the IEC 62264, i.e., Enterprise, Site, Area. For each Area, the corresponding *Logical Devices* and *Logical Nodes* are defined (i.e., the devices). Notice that for each *device*, a compatible *agent* is necessary. Therefore, the LogicalDevice and the LogicalNode constitutes the device identifier, thus, for every LogicalNode within a LogicalDevice an agent is introduced (*4: setAgent(LogicalDevice, LogicalNode, Agent)*). The user is the responsible to introduce

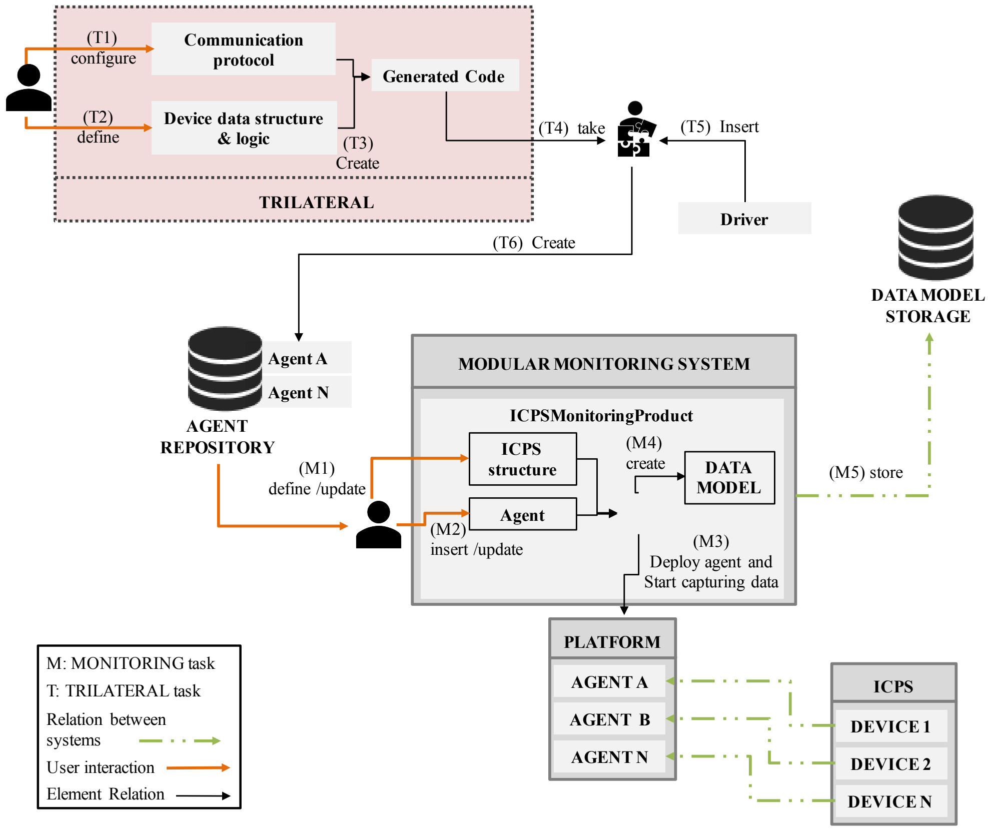


Figure 4.10: Modular Monitoring System Configuration and TRILATERAL

the agent and the *ICPSMonitoringProduct* is responsible for storing it (*5: storeAgent()*. Then, the user needs to specify where (in which platform) the agent is going to be deployed (*7: addAgent2Platform()*).

Once the *ICPSMonitoringProduct* is configured, the user can start monitoring the configured ICPS (*8: startMonitoring()*). Each agent is traced to a binary file that encapsulates the monitoring and/or controlling function dedicated to a certain type of device and it is automatically deployed in the corresponding platform (*9: deployAgents()*). Then, every configured agent is instantiated (*10: createAgentIntance()*), one instantiation per device. In that moment the devices start sending data via agents (*12: sendData()*).

When the *ICPSMonitoringProduct* receives the data through the agent (*13: sendData()*), the *ICPSMonitoringProduct* is responsible of joining the data coming from the agent (device data structure based on IEC 61850) with the organizational structure (IEC 62264), i.e., it is the responsible of creating a concrete *Data Model* considering the *Data MetaModel* (*14: fillDataModel()*). Then as last step the concrete *Data Model*

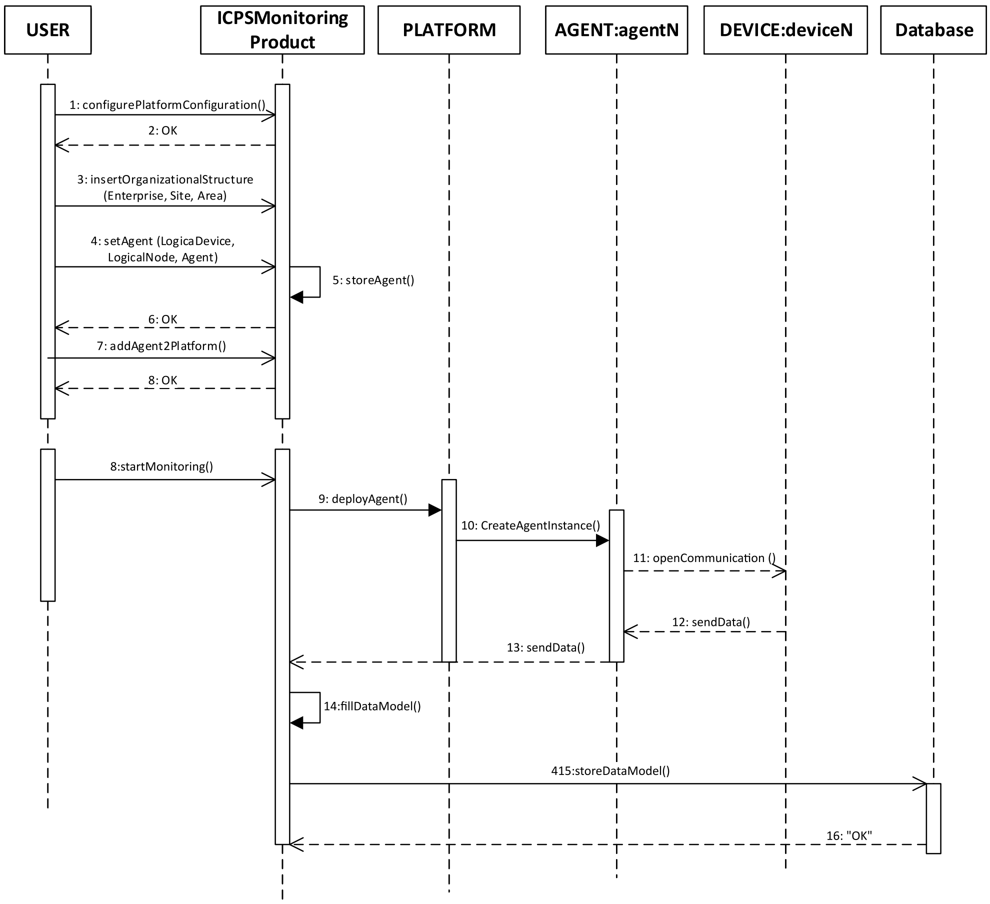


Figure 4.11: Sequence diagram of how a software monitoring system is configured and how it works

is stored in order to have the traceability of the ICPS over time (*15:storeDataModel()*). Note that for the *ICPSMonitoringProduct* the data received from the agent is unknown. Thus, if the agent starts sending new data which was not sent before, *ICPSMonitoring-Product* is going to be able to map that new data and create the corresponding *Data Model*. In order to create a *Data Model*, the *LogicalDevice* and *LogicalNode* are used, i.e., with the *LogicalDevice* and *LogicalNode* the IEC 61850 and the IEC 62264 are joined.

Thus, depending on the agent logic the composition of the Data Model can be different time to time. In addition, the configuration of the *ICPSMonitoringProduct* can vary at runtime without stopping it, e.g., the user can update an agent and the *ICPSMonitoringProduct* does not need to be stopped, this is because each agent is independent.

Therefore, with *modular monitoring system* different *ICPSMonitoringProduct*s can be configured, each one is different since they are configured according to customers' requirement, i.e., it is the user who manages all configurations and changes. The user decides which *devices* to monitor and is responsible for introducing or removing *agents* and/or *platforms* manually in the *ICPSMonitoringProduct* while the *ICPS* is in operation.

Thus, once the *ICPSMonitoringProduct* is configured, and this one starts receiving data from the *devices* a specific *Data Model* is conformed. This *Data Model* reflects an snapshot of an *ICPS* at a particular moment in time using a combination of two different standards (i.e., IEC 62264 and IEC 61850) to obtain structured data.

In conclusion with *modular monitoring system*: all changes are made at runtime without the need to stop the entire system; and the data sent by the devices via agents are automatically structured based on the *Data MetaModel*, i.e., for every timestamp a *Data Model* is automatically created and stored.

4.5 Conclusion

In this chapter, a *modular monitoring and adaptable and adaptive visualization solution for facing the evolution of Cyber-Physical Systems* is proposed which is composed by two main systems: (I) the *modular monitoring system* and (II) the *personalized visualization & evolution detection system*. More precisely, in this chapter the first system is presented, i.e., the *modular monitoring system*, where a specific monitoring system can be configured (*ICPSMonitoringProduct*) by the user. Then the system automatically captures data from the devices using agents to then store that data in a structured way. The *modular monitoring systems* is based on the combination of different international standards. The main challenges of the *modular monitoring system* are that: (I) it is able to monitor different ICPSs, with different ICPS organizational structure, and that the ICPS to monitor can belong to different industrial domains. (II) It is able to handle ICPSs evolution at runtime.

The *modular monitoring system* is composed by three main components: (I) the ICPS to be monitored needs to be structured based on the ISA-95 standard. (II) The communication between devices and the *ICPSMonitoringProduct* is based on Agent-based communication, which make each device autonomous, making possible the ICPS evolution. (III) In order to be aware of what is happening in the real wold

Table 4.3: Achieved requirements by the Modular Monitoring System

Requirements	***Modular Monitoring System***
Monitoring System	■
Communication protocols	■
Evolution Detection	
Adaptable Visualization	
Adaptive Visualization	
Structured Data	■

every timestamp the *ICPSMonitoringProduct* is able to create a concrete Data Model based on a Data MetaModel. The Data MetaModel is a data Representation by IEC 61850 and IEC 62264 standards.

In conclusion, thanks to the *modular monitoring system* it is possible to start monitoring different ICPSs whether they are from the same domain or from different domains. Additionally, the *modular monitoring system* is able to manage ICPS evolution. However, the *modular monitoring system* is no able to face all domain requirements in order to fulfill a *modular monitoring and adaptable & adaptive visualization solution for facing the evolution of Industrial Cyber-Physical Systems* (see Table 4.3). Detecting and alerting about ICPS evolution or automatic visualization adaptation and customization are two requisites that need to be solve, hence, in the next Chapter the *personalize visualization & evaluation detection system* is fully explained.

Chapter 5

Personalized Visualization & Evolution Detection System

This Chapter[1] presents a *personalized visualization & evolution detection system*. The system is capable of (I) detecting and classifying the evolution and (II) alerting to the user about the occurred evolution besides giving the chance to reconfigure User Interface (UI) during operation.

First, the main components of the designed system are presented. Then the workflow of the system is presented, i.e., how to create Graphical Element (GE) by the user, how to detect and classify Industrial Cyber-Physical System (ICPS) evolution and how to adapt the created GEs considering the occurred evolution. The chapter is finished with conclusions.

Contents

5.1 Introduction **84**

5.2 Personal Visualization & Evolution Detection System Components **85**

5.3 Creating GEs at Runtime by the End User **90**

5.4 Automatic Detection and Classification of ICPS Evolution . . . **91**

5.4.1 Data Model Comparator 92

5.4.2 Evolution Classification 94

5.5 Adapting Dashboards to users **96**

5.6 Conclusion **99**

[1]The content of this Chapter has been published partially in [IYA+18, ISA19]

5.1 Introduction

An ICPS needs to be monitored, but it evolves over the time. Therefore, aside from updating the monitoring system, the UI needs to be updated. As mentioned in the previous section, three more requirements need to be achieved to fulfill a *modular monitoring and adaptable & adaptive visualization solution for facing the evolution of Industrial Cyber-Physical Systems*: (I) evolution detection, (II) adaptable visualization and (III) adaptive visualization.

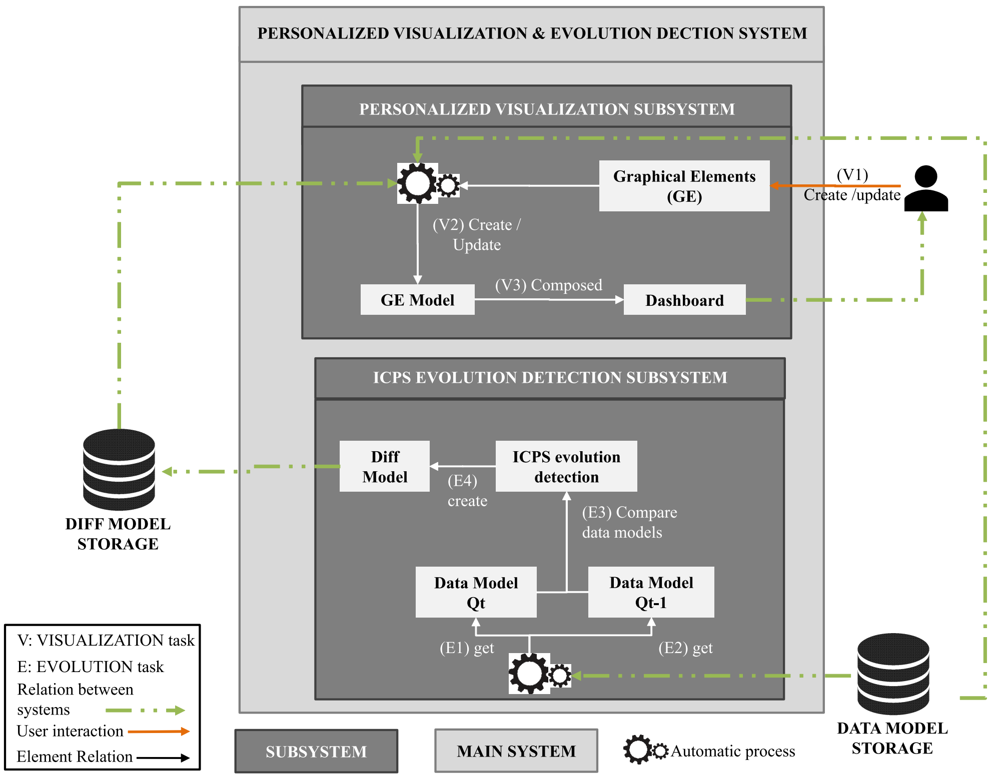


Figure 5.1: Personalized visualization & evolution detection system overview

To cope with these requirements, two subsystems were defined within the *personalized visualization & evolution detection system* (see Figure 5.1). Firstly, the *ICPS evolution detection subsystem* is responsible for detecting and classifying ICPS evolution automatically. Secondly, the *personalized visualization subsystem*, is a semi-automatic system responsible for creating GEs with the guidance of the user and adapting those GEs if any evolution is detected, as long as the GE needs to be adapted. Thus, the presented system:

- Allows users to create new visualizations or update existing ones at runtime, and share visualizations with others considering data captured from an operating ICPS, i.e. *ICPSMonitoringProduct*.
- Detects and classifies ICPS evolution automatically.
- Updates the UI created by the user considering evolution and informing users about the changes.

5.2 Personal Visualization & Evolution Detection System Components

In this section the general concepts of the designed system are presented. Figure 5.2 presents the main components of the *personalized visualization & evolution detection system*. With this system, each user will have her/his dashboards based on her/his criteria. Using such a structure, even if the ICPS under monitoring is the same, two different users can visualize the same data in different ways or even different data.

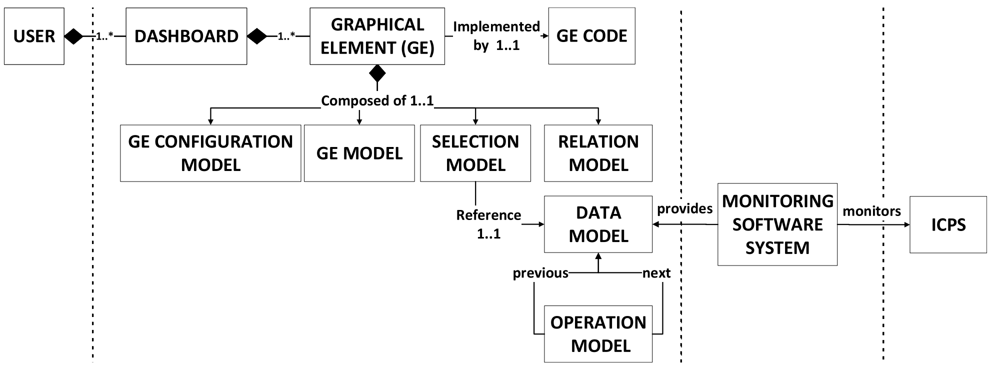


Figure 5.2: Main Components of the *Personalized visualization & evolution detection system*

Every dashboard is composed of different GEs (UIs components), and the user designs the composition of each dashboard. Considering that a GE needs to be configured by an user and automatically updated by the *personalized visualization & evolution detection system*. By combining the information captured from different models, you get a better runtime adaptation, since the output of one model is the input of others, which gives it more flexibility to adapt to the user [BBCW18]. Thus, each GE is created by combining different models and each model is instanced from a concrete MetaModel. Every model, provides different information about the GE :

- **Data MetaModel**: It is the model that stores the data captured from the ICPS (by *ICPSMonitoringProduct*) with a specific structure. The Data Model diagram is shown in Figure 4.7 on Section 4.3.3.

- **Selection MetaModel**: It stores the concrete data structure that the end user wants to visualize, i.e., Which data is visualized?

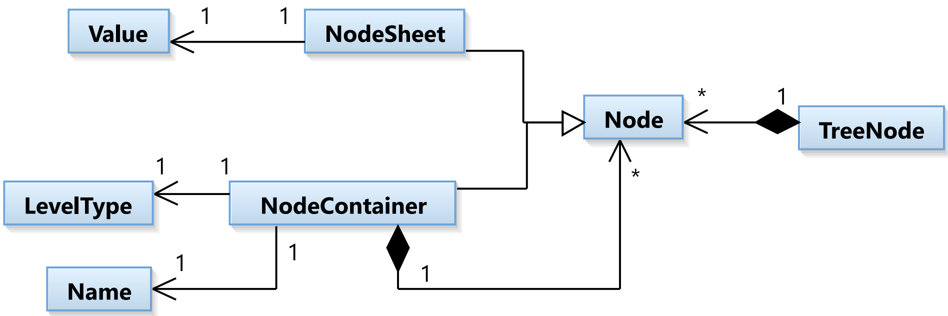


Figure 5.3: Selection MetaModel

This model is based on the Data Model and the main difference is that the number of levels is not fixed (i.e seven levels composed by Enterprise, Area, ...), so the tree levels will depend on the selected data to create the GE. Thus, as shown in Figure 5.3, the Selection Model has a tree structure (*TreeNode*) that it is composed by one to many *Nodes*. Two types of *Nodes* exist:

- *NodeSheet*: It is the last node of the tree, thus, it would be composed by a value.

- *NodeContainer*: It is a node which can have more *Nodes* inside it and each *Node* will have a name and a *LevelType*. The *LevelType* is the one who provides information about the level of the *Data MetaModel*.

- **GE Configuration MetaModel**: It captures all the information about the appearance of the GE (PieChart, Table, BarChart), i.e., What type of GE is displayed?

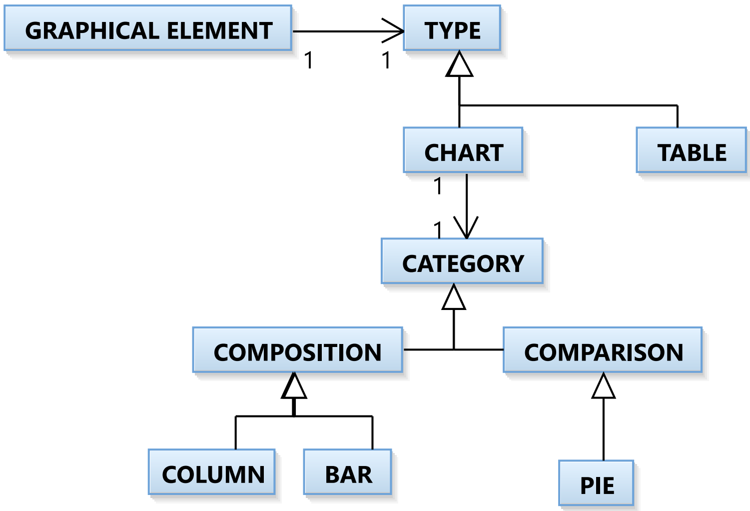


Figure 5.4: GE Configuration MetaModel

In Figure 5.4 the configuration of a GE is reflected. As shown in the figure, every GE belongs to a *type* (e.g., chart, table). Depending on the GE *type*, the configuration of each GE is different. For example, a bar chart, belongs to a specific *category*, i.e, composition, which in turn belongs to a concrete *type*, i.e., chart. Note that, this MetaModel can evolve, since depending on the chart, tables, etc. that the user wants to use every, MetaModel needs to be adapted by the developer.

- **Relation MetaModel**: Taking into account the selection made by the user about the data to visualize and the GE configuration, it is possible to create the *Relation model*, i.e., it is the join of the GE and the data to visualize. Therefore, apart from configuring the GE (*GE Configuration Model*), the user needs to configure the relation between the GE and the Data to visualize. To do so, s/he needs to specify: (I) *Action*, the activity that the GE is going to visualize (e.g., count the quantity of "something"). (II) *Action To*, it refers to the *Level Type* in which the *Action* is going to be applied. 3) *Group by*, using this artifact, the user can group the nodes to be displayed if this is of interest, e.g., group *LogicalNodes* by existent *Areas*.

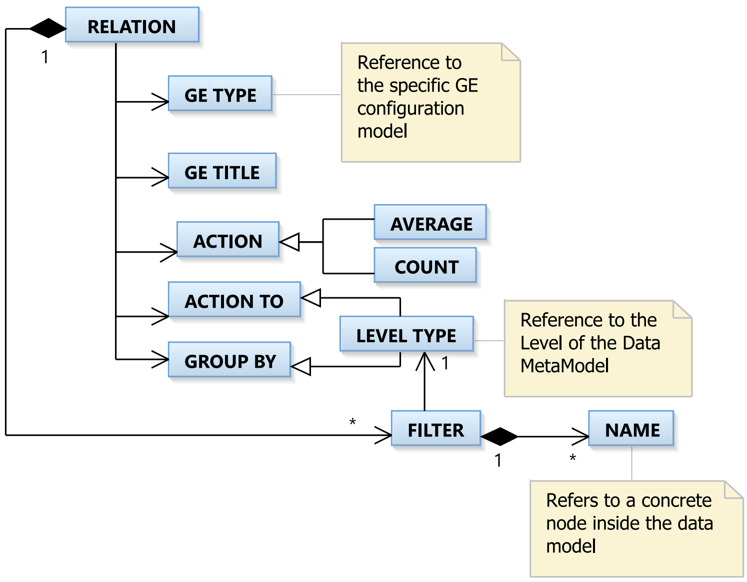


Figure 5.5: Relation MetaModel

- **GE MetaModel**: It interprets the *relation model* an creates the corresponding code in order to visualize the GE in the Dashboard to the end user. As shown in Figure 5.6, a *user* can configure one to many dashboards. Each GE can belong to more than one *dashboard*.

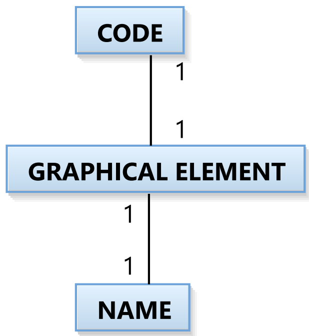


Figure 5.6: GE Model

- **Operation MetaModel**: The Operation Model contains information about the types of changes, the level in which changes have occurred and the nodes affected by the changes. The *level type* is related to the level of the Data MetaModel and describes in which level of the Data Model the changes occurred (e.g. site).

Inside each level type, the MetaModel considers three types of changes that occur when an ICPS evolves (see Figure 5.7).

- *ADD*: All the new nodes which did not exist in the previous instant.
- *REMOVE*: All the nodes that have been deleted.
- *MODIFY*: If the node exists but a change has been detected.

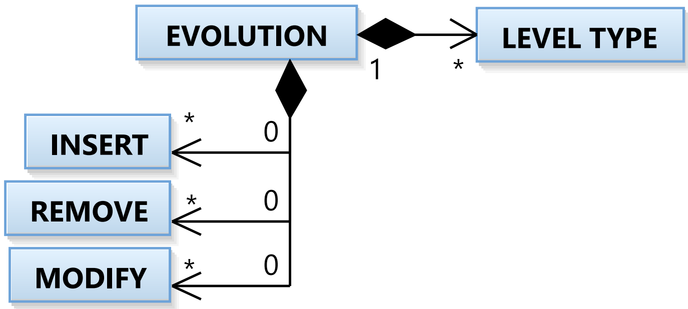


Figure 5.7: Operation MetaModel

Table 5.1: Models dependent on users or ICPS evolution

■ Yes □ No

Reference	Model name	*User update*	*ICPS evolution*	Depends on
M1	Data Model	□	■	-
M2	GE Configuration Model	■	□	-
M3	Relation Model	■	□	M1; M2

The MetaModel instantiation (a concrete model) changes over time due to different reasons: (I) a change made by the user (e.g., a GE composition is updated), (II) evolution of the ICPS (e.g., a new attribute has appeared, thus, a concrete model needs to be updated) or (III) a change in another model on which a model depends on. Depending on the change, the model to be updated will be different. This will make its maintenance easier since the changes only affect one model at a time. The rest of the affected models will change automatically without user interaction. In this manner, it is possible to create the corresponding *GE Code*, i.e., the code used in order to display the GE to the end user in the Dashboard. Table 5.1 describes each model along with their relationships, i.e., which model can be updated by the user and which ones are automatically updated by the ICPS evolution.

Table 5.2: Dependencies between models

Reference	Model name	Depend on
M4	Selection Model	M1, M2, M3
M5	GE Model	M2, M3
M6	GE Code	M5
M7	Operation Model	M1

Some models are dependant on other models, i.e., if model changes (e.g., M2: GE Configuration Model), this can trigger another model to change (e.g., M4: Selection Model). These dependencies are shown in Table 5.2.

5.3 Creating GEs at Runtime by the End User

This Section describes the *personalized visualization subsystem* that allows the users to manage and create their own visualizations. Figure 5.8 summarizes how a new GE is created using the designed system in collaboration with the user.

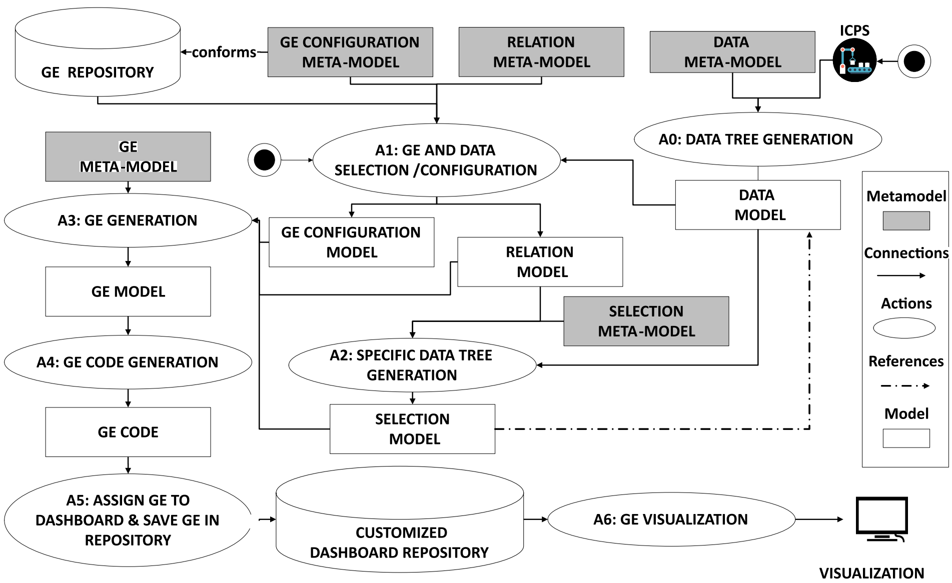


Figure 5.8: User Creating a New GE

Once data is stored in the *Data Model* (A0), in order to create a new GE, the user interacts with the system (A1). First, the user needs to configure the GE (*GE*

Configuration Model). Note that configuring different GE types might be different, implying that the *GE Configuration Metamodel* should be different for each GE type. Then, the user needs to select which information to display (*Relation Model*). The relationship between the configuration of the GE (*GE Configuration Model*) and the data to be displayed is stored in the *Relation Model* different for every GE.

For example, suppose the user wants to visualize the data in a multi-level pie chart, i.e., a pie chart. The pie will be the GE that the user wants to configure and this GE will have its *GE configuration MetaModel*. The user will configure the GE in order to create model (*GE configuration Model*) that conforms to the MetaModel of the pie chart. Then, the user selects the data to visualize in that GE (*Relation Model*), i.e., which information to display on the pie chart. Then, taking into account the *GE Configuration Model*, the *Relation Model* and the *Selection MetaModel*, the *Selection Model* is created.

The *Selection Model* is a set of the Data Model for a concrete configuration done for the user. The *Selection Model* references to the Data Model, therefore, data duplication is avoided and every GE only usages the necessary data.

The *GE Model* is created automatically (A3), with the information captured by the *GE Configuration Model*, *Relation Model*, and *Selection Model*. The system uses this model for generating the corresponding code to display the GE to the user, i.e., GE Code (A4). Technically, the GE Code can be created by existing libraries such as HighCharts[2], D3[3]. The last step is (A6) to assign the created GE to an existing or a new dashboard (A5).

5.4 Automatic Detection and Classification of ICPS Evolution

As mentioned above, the ICPS can evolve due to different factors, i.e., new elements are introduced, or existing ones are removed or replaced; or new attributes are received from an agent. In this section focuses on the *evolution detection and classification subsystem*, i.e., a subsystem that compares the current instance (Data Model Qt) with the previous instance in time (Data Model Qt-1) to identify the evolution of an ICPS.

[2] https://www.highcharts.com/
[3] https://d3js.org/

The objective of this system is to detect an ICPS evolution. Our solution is able to identify at any level, additions, removals or modifications by comparing Data Models in subsequent timestamps.

This subsystem is composed by two main components: (I) Data Model Comparator, the component responsible of comparing two subsequent Data Models; and (II) Operation Models, the instance of Operation MetaModel responsible of classifying the evolution.

5.4.1 Data Model Comparator

To perform the comparison between the subsequent Data Models, JaVers[4] is used. JaVers is a library able to compare complex structures and detect changes. Despite that, the biggest disadvantage of JaVers is that it does not take into account the hierarchical dependencies between nodes. However, the dependencies in an industrial environment are something necessary, because it is valuable to visualize the result in a simple and meaningful way to the user in order to help him/her making decisions. That is why, JaVers result needs to be post-processed. For example, imagine that due to business strategy in the Press Line (PL) domain it is necessary to remove Zone B. In the upper part of Figure 5.9 the Data Model before an evolution occurred (Qt-1) is shown, e.g., the Press Machine product line of Mexico is composed by two areas. However, in the lower part of the figure, the Data Model after an evolution (Qt) is presented, e.g., Zone B is removed from the Press Machine product line. Therefore, all nodes that depend on that zone are removed (e.g. Machine 1). When our subsystem compares Qt with Qt-1 using JaVers, this one detects a change for each modified, added or removed node. In this case, JaVers generates 1001 alerts when Zone B node is removed using Json Object format or 1502 alerts using Json Array format (See Table 5.3). In the former one, it does not take into account the dependencies between nodes. The latter one, besides not taking into account dependencies, each movement of a node in terms of position (left/right) is considered a change.

This quantity of alerts do not facilitate the task to the user when data is represented. Even if the example given is due to a business strategy, note that as mentioned in Section 3.2, meany reasons can trigger the evolution of an ICPS, e.g., devices' intelligence itself can cause changes in the Data Model.

[4] https://javers.org/

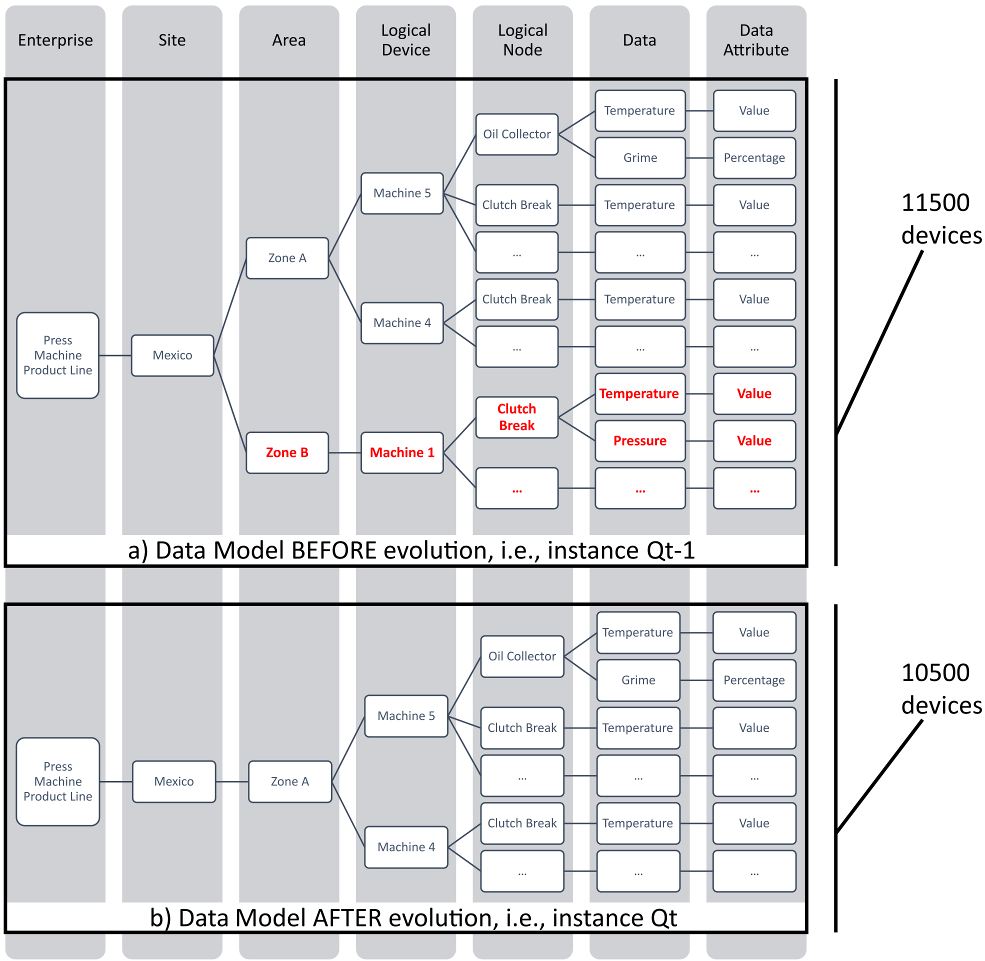


Figure 5.9: Data Model evolution example: a) before evolution, b) after evolution

Thus, the comparison provided by JaVers is format dependent, i.e., the format of the text model impacts the result. If Json arrays [Bo10] are used, the order is taken into account. Thus, when removing node Zone B, nodes on the right are marked as modified, i.e., Zone C is marked as modified, ascending the number of alerts to 1502. Instead, if Json Objects are used [Bo10], when the order of a node changes (due to an addition or removal) it is not marked as a change.

To overcome this limitation, besides using Json Objects, JaVers result is post-processed. Thus, it is possible to only generate the necessary alerts for the user. In the example explained above, a unique alert will be only generated, i.e., ZoneB is removed. Considering the limitation, it was necessary to extend the *Operation MetaModel* to correctly classify the evolution, i.e., Diff MetaModel which is explained in detail below.

Table 5.3: JaVers output depending on the input format

Json Type	**Quantity of Changes**	**Observations**
ARRAY	1502	The movement of zone C to the right is considered a change (500 + 1 changes, zone C and all the devices). The insertion of new elements is considered a change (1000 + 1 new insertions, zone B and all the devices).
OBJECT	1001	The insertion of every node is considered a change (1 + 1000, Zone B and all dependent nodes).

5.4.2 Evolution Classification

In Figure 5.10 a diagram of the process for creating the concrete Diff Model is presented, a concrete instance of Diff MetaModel. The process starts when a new Data Model is received. In that moment, the evolution detection and classification subsystem gets the received Data Model and the previous instance, i.e., Qt and Qt-1. Then, the Data Models are compared by the Data Model Comparator using Javers as explained in the previous section. If the result is not empty, the subsystem gets the result of Javers and handles each action differently (add, remove, change). This is because the information to be saved depends on the type of change that occurred. Once the attributes are mapped, the information is classified based on the Data MetaModel types (see Chapter 4 Section 4.3.3 Table 4.1). This manner simplifies to identify where the evolution has occurred. Then, if elements are added or removed, it is necessary to delete the unnecessary information despite of duplicated information as mentioned above (e.g. from 1001 alerts to 1, i.e., it is not taken into account nodes below Zone B). Finally, all the information is set to create the Diff Model. Once the process finishes, the created Diff Model is stored in a NoSQL database in order to keep track of the traceability of the evolution over time.

As mentioned above Diff MetaModel is an extension of the Operation MetaModel that contains information about the types of changes, in addition to the classification of the ICPS evolution. In order to communicate to the end user in a fast and efficient way the evolution, all the identified changes in a level (e.g., area) are grouped. Therefore, every Diff Model instantiated from the Diff MetaModel, can have a maximum of seven level types, one for each Data MetaModel level.

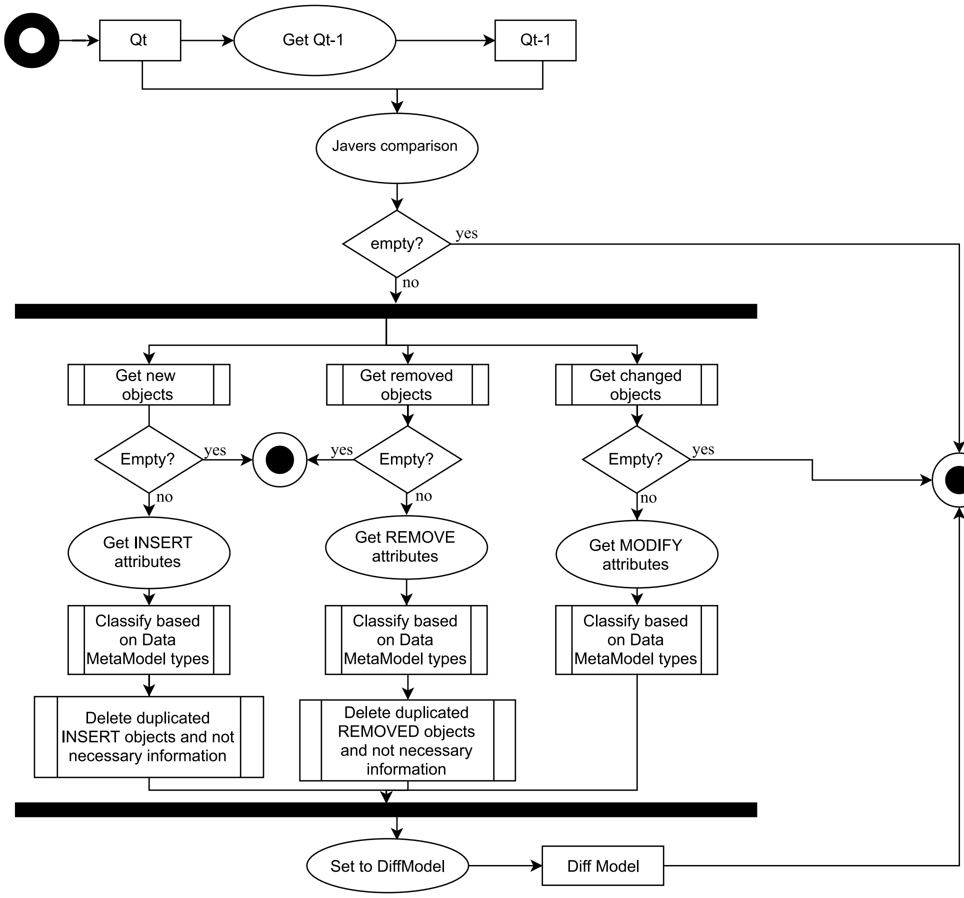


Figure 5.10: Diff Model creation diagram

To reduce the quantity of alerts (i.e., *delete duplicated INSERTED/REMOVED objects and not necessary information*; see function of Figure. 5.10), e.g., from 1001 alerts to 1 alert, a FatherChildNode is used to group the affected nodes, see Figure. 5.11. The FatherChildNode is an instance of a DataModelChild, that is, a subtree with the node changed and its children and contains the nodes affected by addition and removal. Note that the FatherChildNode does not need to contain the seven levels (i.e. enterprise, site, ...).

When a node is removed or inserted, a FatherChildNode is saved that extends from GroupElement i.e., the GroupElement describes the FatherChildNode since it describes the DataModelChild. For example, Figure 5.9 only shows the first five levels (enterprise, site, area, logical device, and logical node), in a real scenario will be composed of seven levels, when Zone B is removed, the FatherChildNode would be composed by Zone B and its children, i.e., DataModelChild will only contain five levels (area, logical device, logical node, data and data attribute), enterprise and site

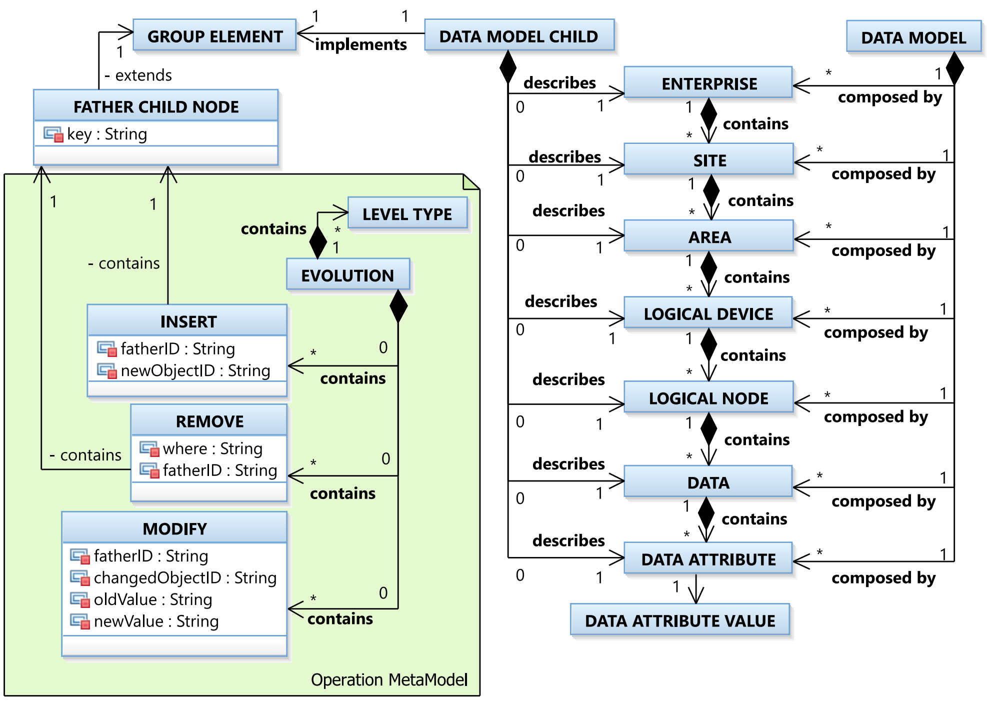


Figure 5.11: Diff MetaModel Structure for alerting users about ICPS evolution

will not be saved. In this manner, the number of alerts is reduced. A description of each variable in the Diff MetaModel is presented in Table 5.4.

A Diff Model that conforms to a Diff MetaModel, is created automatically from JaVers output every timestamp and then it is stored in a NoSQL database. Thus, it is possible to classify all the changes so that they can be presented easily to the user. Thanks to this MetaModel, the number of alerts are reduced and, redundant information is avoided because it is just saved the deepest node that is changed excluding all below nodes.

5.5 Adapting Dashboards to users

Once explained how an ICPS evolution is detected and classified, in this section how the system is able to communicate the evolution to the end user is presented. Figure 5.12 summarizes the process of how the *personalized visualization & evolution detection system* updates GEs when an ICPS evolution occurs. Thus, when a new DataModel is received, as first step (OP1), the system compares the *«current» Data Model* (Qt) with the *«previous» Data Model*.

Table 5.4: Description of the information saved in each Diff Model

Variable	**Description**
father_ID	The identifier of the node from which an element has been added, deleted or changed.
newObjectID	The identification of the newly inserted node
where	A path pointing to the entire chain from the enterprise to the newly inserted object.
FatherChildNode	The tree that depends on the inserted or deleted node. This tree no necessary has seven levels, it will depend on the fatherID level.
changedObjectID	The identification of the changed node
oldValue	The value previously held by that node
newValue	The value is currently held by the node

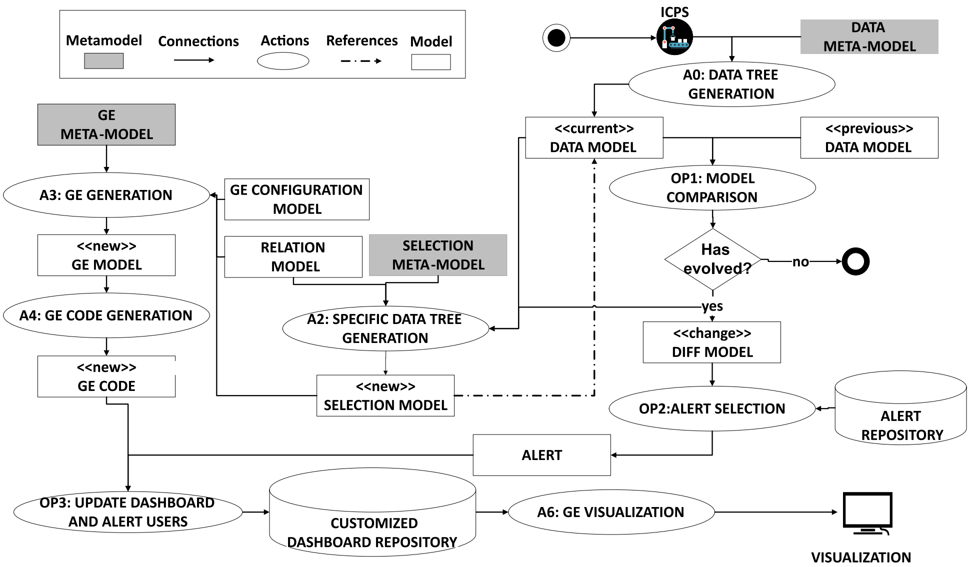


Figure 5.12: Managing ICPS changes at runtime

Once the *Diff Model* is created, for every change that has occurred, an alert is generated (OP2). In this manner, the user is informed about every evolution that has occurred in the ICPS and therefore has the complete overview of the ICPS.

Users will receive different alerts depending on the changes happening in an ICPS (see Figure 5.13):

- **Warning Alerts:** The GE is not affected, but there has been a change, which is somehow related to it.
- **Information Alert:** The GE must be updated due to a structural change.
- **Error Alert:** It is not possible to visualize the GE correctly, due to ICPS evolution.

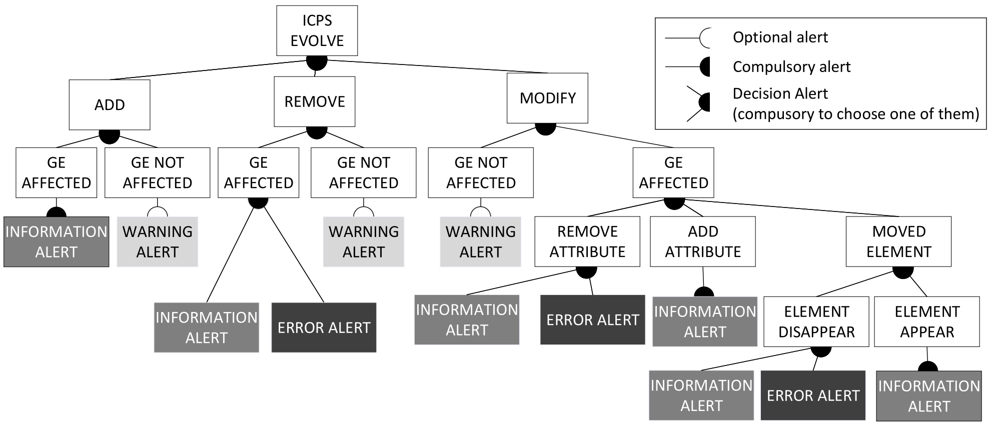


Figure 5.13: Type of Alerts a user can receive depending on ICPS evolution

Besides selecting the corresponding alert for each change, the *«new» Selection Model* is created with the information stored in the *«current» Data Model* and the *Relation Model* already configured by the user.

Before informing users about changes, the GE is updated (OP3). Using the *GE Configuration Model*, *Relation Model, and «new» Selection Model*, the *«new»GE Model* is created. Thus, it is possible to generate the *«new» GE Code* for generating the updated GE. Then, the affected users are searched, so all of them are informed about changes that occurred in the ICPS. Finally, the GE is updated, and the corresponding user is informed. If the GE (e.g., multi-level Pie Chart) can be modified, the *personalized visualization & evolution detection system* adapts the GE to the configuration chosen by the user (*GE Configuration Model*, *Relation Model*). Otherwise, it will alert (error alert) the user that it is not possible to generate the GE.

Note that a user can visualize more than one GE in a dashboard. Thus, more than one GE can be updated when the ICPS evolves. Also, one GE can be affected by different changes in the ICPS. Thus, for every updated GE, the user will receive at least one notification. In the same way, when the ICPS evolves, the UI is updated automatically at runtime by the *personalized visualization & evolution detection system*.

In the same way that the ICPS evolves, user visualization requirements can also change. Therefore, our system provides the opportunity to (I) change the configuration, (II) modify the data to be displayed, or (III) make both changes to an existing GE. In the former, the user needs to update the already made selection, and the *personalized visualization & evolution detection system* will automatically change the configuration. The user must then select the data to be displayed, taking into account the new configuration selected. Thus, the *GE Configuration Model*, besides, the *Relation Model* will be updated. In the latter, after selecting the new data, the *Relation Model* will be updated by automatically by the system. Then in both cases, the *Selection Model* is automatically updated by taking into account both models aforementioned. Subsequently, the system will search for users who are visualizing the updated GE and notify them of the change that occurred in addition to updating the GE.

5.6 Conclusion

The need to adapt user interfaces when an ICPS evolves is becoming a mandatory requirement for several industrial domains, such as, Automated Warehouse (AW), PL and Catenary-Free Tram (CFT) domains. After conducting a domain analysis of these domains the requirements of (I) detecting evolution, (II) creating adaptable and (III) adaptive visualizations were identified. Thus, as shown in Table 5.5, the *personalized visualization & evolution detection system* was designed in order to address the identified requirements to fulfill a *modular monitoring and adaptable & adaptive visualization solution for facing the evolution of Industrial Cyber-Physical Systems*.

Due to the designed architecture, it is possible to manage UIs, users will have more flexibility and will be able to create their own GE without developers help. In that way, controlling the ICPS and decision making (e.g., introduce a new device in the ICPS) will be easier, cheaper, and faster because the system itself will help the user on visualizing the needed information and creating alerts if it is necessary. The user will be responsible of generating the GE. Also, to give a chance to the user to configure her/his own GE, sharing those with teammates is useful. Once the GE is created, other users can use it without having to configure a new one and, thus, further facilitating discussion.

Table 5.5: Achieved requirements by the *personalized visualization & evolution detection system*

Requirements	**Personalized Visualization & Evolution Detection System**	
	Personalized Visualization Subsystem	**ICPS Evolution Detection Subsystem**
Dynamic Monitoring System		
Communication		
Structured Data		
Evolution Detection		■
Adaptable Visualization	■	
Adaptive Visualization	■	

PART IV

DEVELOPMENT & EVALUATION

"The moment when you want to quit is the moment you need to keep pushing"

ANONYMOUS

Chapter 6

Prototype & Evaluation

This Chapter[1] presents the developed prototype and the evaluation of the two proposed systems, i.e., the *modular monitoring system* and the *personalized visualization & evolution detection system*. In addition, how the systems can be applied into other domains is presented.

First, how the systems were developed and how they were tested is presented. Next, the return of investment of both systems are presented, followed by a user based survey where the interest of the developed systems are evaluated. The Chapter is finished with a scalability test where it is evaluated if the presented system is scalable to different domains. The chapter is concluded with the presentation of what is necessary to know (experts point of view) in order to apply the system into other domains.

Contents

6.1 Prototype & Validation **104**
- 6.1.1 Modular Monitoring System 105
- 6.1.2 Semiautomatic Agent generation 108
- 6.1.3 Personalized Visualization & Evolution Detection System 112

6.2 Evaluation **116**
- 6.2.1 Return Of Investment 116
- 6.2.2 Survey about proposed solution 120
- 6.2.3 Scalability test 125

6.3 Applying Solution into other domains **134**

[1]The content of this Chapter has been published partially in [IYA+18, ISA19, IIL+19a, IIL+19b]

6.1 Prototype & Validation

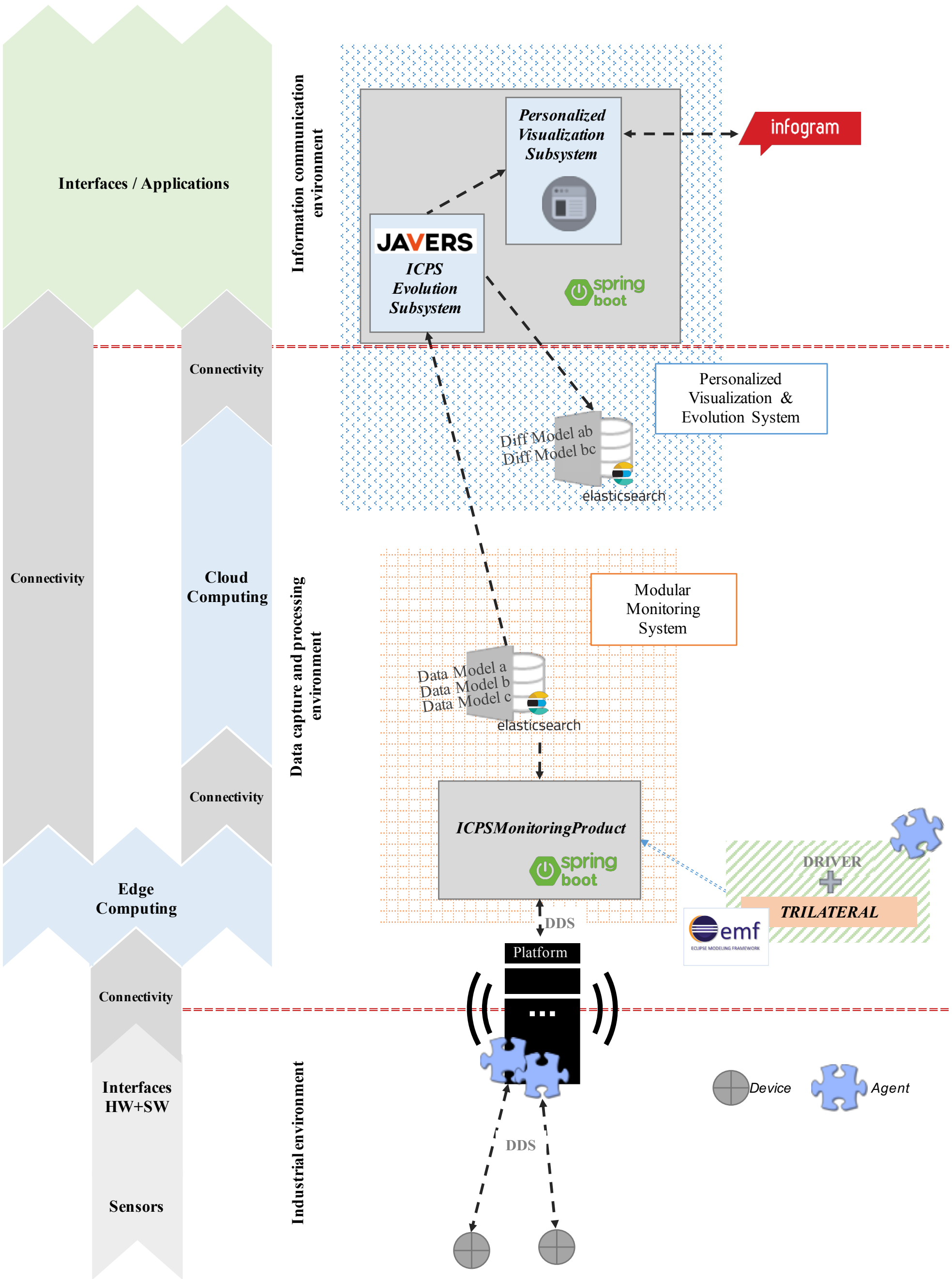


Figure 6.1: Prototype architecture

The designed solution is based in two main systems: (I) a *modular monitoring system* for evolving Industrial Cyber-Physical Systems (ICPSs) and (II) the *personalized* *visualization & evolution detection system*. Thus, in this section a prototype of the solution is presented in order to prove the technical feasibility. In Figure 6.1 a unified prototype is presented, in spite of it, each system was developed and validated individually.

First, in Section 6.1.1 the technologies used to develop the *modular monitoring system* and its validation are presented followed by Section 6.1.2, where how sof**T**ware p**R**oduct l**I**ne based mu**L**tidomain iot **A**r**T**ifact g**E**neration for indust**R**i**AL** cps (TRILATERAL) was developed is explained. Finally in Section 6.1.3 how the *personalized visualization & evolution detection system* was developed is presented.

6.1.1 Modular Monitoring System

In order to validate the feasibility of the proposed system, a prototype of the *modular monitoring system* was developed. Then with the prototype a concrete *ICPSMonitoringProduct* was configured for ULMA Handling Systems in Automated Warehouse (AW) domain. The developed system was also used for MegaM@Rt2 project[2].

In order to start monitoring an ICPS which is composed by different devices, one agent per device needs to be developed, i.e., each device has each own agent (*BusinessAgent* or *PLCAgents*) in order to communicate the devices with the monitoring system, i.e., *ICPSMonitoringProduct*.

Each agent has a *communication protocol* and the *data structure & logic*. In this particular case the communication protocol used was Data Distribution Service (DDS) [OMG15] since it enables easier communication when a high quantity of different data is required to be transferred [PKKP10]. Considering this prototype was supervised by ULMA, it was not possible to create the agents with TRILATERAL, since this communication protocol was not available.

For our prototype, two agents were created. They both were scales but the data sent by each one was different, i.e., the logic of the agent was different: one agent was able to send the average of the last 15 measures, instead the other one no. The agents were developed in Java-programming language and every agent has a *data structure & logic* and a *communication protocol*. In this particular case, the *data structure & logic* code of each agent is about 170MB and the *communication protocol* about 800KB.

[2] European Union's Horizon 2020 research and innovation programme under grant agreement no. 737494

Apart from developing the necessary agents, the *modular monitoring system* was developed in order to configure the concrete *ICPSMonitoringProduct*. The *modular monitoring system* was developed in Java-programming language and it makes use of SpringBoot library. The source code has 430KB which is able to: (I) configure the organization structure, (II) launch the agents in the corresponding platforms, and (III) run or stop agents in the platforms.

In order to validate the created prototype, an AW monitorization was simulated. First the platforms were configured. To do so, in each platform a specific software is launched manually; this software has about 75KB. This specific software is capable of receiving the binary of a concrete agent and launch it within the platform. A platform has a common file that is able to download the agent and launch it in order to start capturing data, and a personalized file where the identification of the platform is specified. Thus, when more than one platform exist, the *ICPSMonitoringProduct* knows to which platform the agent should send and from which platform it is receiving data thanks to the configuration made by the user.

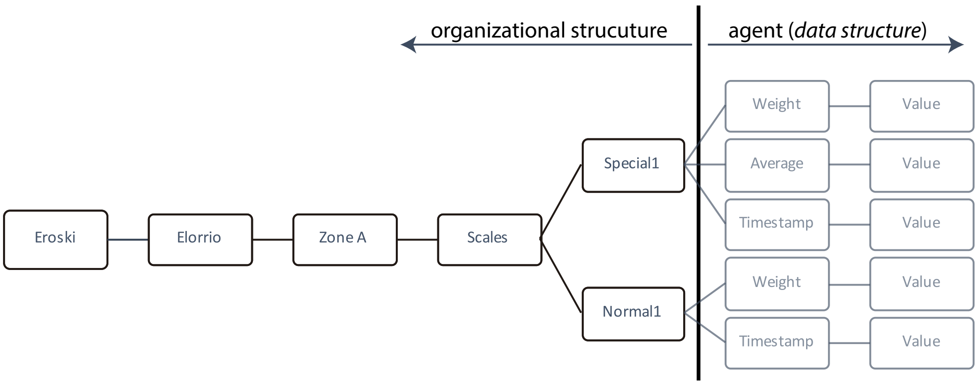


Figure 6.2: Configured ICPS structure

Next, the organizational structure is defined, i.e., in this case two agents were introduced in the *ICPSMonitoringProduct*. The configured structure is shown in Figure 6.2. The *organizational structure* is the part that the user will introduce in the *ICPSMonitoringProduct*. The *data structure* is configured when the agent is created. Finally, the user gives the order to the *ICPSMonitoringProduct* to start monitoring the ICPS. In that moment, these agents are deployed automatically in the corresponding platform and the devices start sending data via agents. Then, the agents start to send the corresponding data to the *ICPSMonitoringProduct* using the International Electrotechnical Commission (IEC) 61850 data structure. In this particular case, the communication protocol used in each agent was DDS. When the agents communicate with *ICPSMonitoringProduct*, this one completes the corresponding Data Model for every timestamp, i.e., the data sent by the agent is merged with the organizational structure defined by the user. This is possible due to the device identifier, i.e., the device identifier is a combination between the LogicalDevice and the LogicalNode. In this way, the system can combine the data coming from the agent with the organizational structure already defined by the end user. Once the data model is conformed, this one is stored. There are many formats and databases where data can be stored. In our case, a NoSQL database is used. There are many reasons for using NoSQL databases [TAS12]: (I) they handle data with different structures and (II) they are highly scalable and reliable. Such characteristics are relevant in this context considering the ICPS can evolve since it is necessary to handle unstructured data since our Data Model will evolve from time to time. Additionally, as mentioned in the performed domain analysis different sizes of ICPSs exist (e.g. medium, large, small), thus, depending on the ICPS size to be monitored the received data can be large.

Thus, in our case, ElasticSearch is used and the document format is JSON format in order to save the tree structure in the NoSQL database.

In order to validate that the prototype works, some experiments were performed in order to satisfy the requirements defined:

- Question 1: Is the Data Model correctly created?
  - Action: Define an ICPS structure.
  - Check: The expected Data Model is correctly stored in the database.
  - Result: Correct.
- Question 2: Is possible to introduce an agent at runtime and receive the data automatically?
  - Action: Update the organizational structure, introduce the agent and give the order to start monitoring it.
  - Check: The Data Model is correctly created with the corresponding structure.
  - Result: Correct.
- Question 3: Is possible to receive device data with different protocols?
  - Action: Configure agents with different communication protocols.

- Check: The expected data is correctly received.
- Result: Not evaluated. The agents can be created with different protocols as evaluated in next section and in [IIL+19b], but the monitoring system was not evaluated in terms of receiving data in different protocols.

- Question 4: Is possible to receive unknown or not expected data?
  - Action: A "new" attribute is stored correctly in the database.
  - Check: The alert and the corresponding visualization are correctly created/updated.
  - Result: Correct.

6.1.2 Semiautomatic Agent generation

TRILATERAL is a graphical eclipse plugin developed with the Eclipse Modeling Framework[3]. The *information model*, i.e., the tree structure of the ICPS to be monitored/controlled, can be represented with TRILATERAL. Figure 6.3 shows a screenshot of TRILATERAL, where a *Server* named *SERVER_NODE* is shown. The *Server* has the *ClimateSystem_LD LD*, which includes several *LNs*. Those *LNs* then have different *Datas*, *Datasets* and *CBs*.

As mentioned in previous Section, the code is generated automatically from the designed *information model* with TRILATERAL. Additionally, the user can choose one or more protocol between WS-SOAP, HTTP-REST or CoAP. The code which is generated composes several Eclipse projects, in three different layers, i.e., kernel, libraries and applications. The layers of projects and libraries are shown in Figure 6.4. The kernel of the agent is in the lower level. It includes the lib-model-kernel for the generic parts of the model along with the lib-model-*specific-model*, which is the model generated with TRILATERAL. On top of that (i.e., libraries layer), lib-service-server-rest, lib-service-server-soap and lib-service-server-coap are the libraries to create the servers for HTTP-REST, WS-SOAP and CoAP respectively. Other auxiliary libraries are also in this layer, i.e., libcoap for CoAP communication, libcbor for CBOR information representation, jsoncpp for JSON representation, microhttpd for HTTP-REST communication, and gSOAP for using WS-SOAP. The server for each protocol is on the top layer, i.e., application layer.

[3] https://www.eclipse.org/modeling/emf/

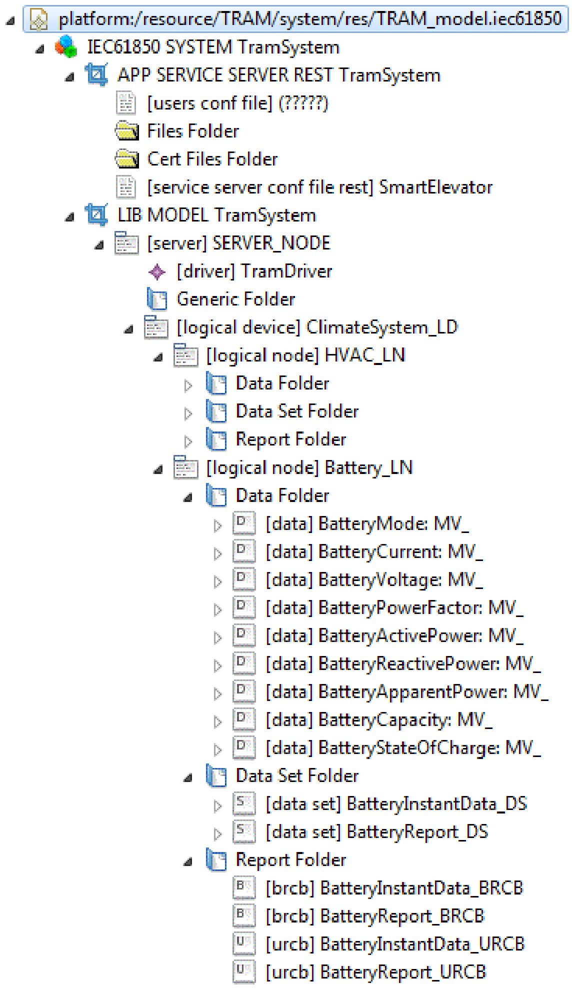


Figure 6.3: Screenshot of TRILATERAL.

Even though the reporting servers are generated apart from the general server, they also have the same three layers. WS-SOAP and HTTP-REST need separate servers and clients because the communication paradigm changes. This does not happen with CoAP due to its *Observe extension* for Pub/Sub communication. Each of HTTP-REST and WS-SOAP reporting server and clients are generated from its own library, namely lib-reporting-server-rest, lib-reporting-client-rest, lib-reporting-server-soap and lib-reporting-client-soap. For reporting functions both WS-SOAP and HTTP-REST work in a similar way, where the service server stores the reports on a folder defined in the system's configuration file. The reporting client has to periodically check if there is any report on that folder, and if so, it sends it to the reporting server. CoAP implementation includes the *Observe* extension to allow Pub/Sub communications, thus, the client just subscribes to the reports and the server pushes them to the client when generated.

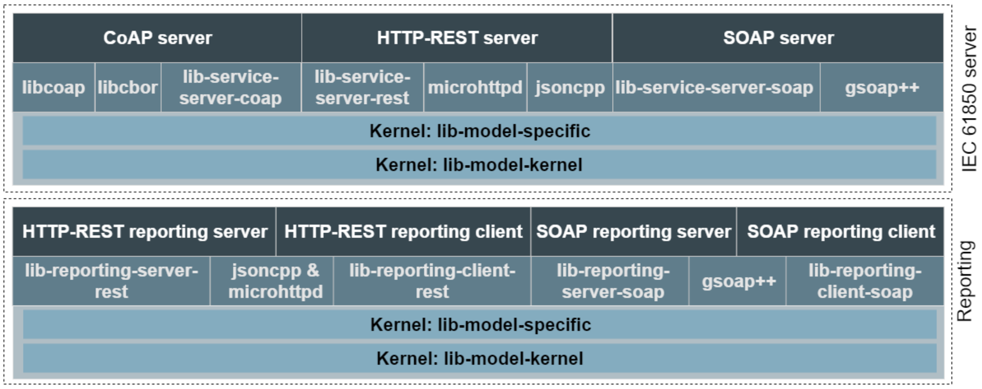


Figure 6.4: Layers of the TRILATERAL tool.

In order to evaluate the tool, TRILATERAL was used to generate first the model and then the source code that allows to construct the set of agents that are deployed on the corresponding platform.

An specific experiment was made for Catenary-Free Tram (CFT). For this, two different scenarios were defined: (I) when the tram is on route, or (II) when it is on a station. Depending on the scenario, the agent needs to communicate with the monitoring system with a different protocol, i.e., for the first scenario (tram is on route), Constrained Application Protocol (CoAP) protocol was used, because it is a mobile scenario where connectivity or power issues can occurred. For the second scenario, i.e, when the tram is on a station, Hypertext Transfer Protocol (HTTP) was used, since the scenario is in a controlled environment, there are not connectivity issues.

In order to validate if TRILATERAL is able to create correctly agents, two different agents were modeled, one to represents the active demand management system and a second one that represent the climate system of the tram. Both of them where modeled by TRILATERAL and two different protocols were used.

An issue arouse on the deployment process, i.e., the agents were too heavyweight. As show in the Table 6.1 the size of the agent can be reduced between a 50% and 30%, as shown in Figure 6.5, if only the necessary protocol is chosen. This led to a change in TRILATERAL, making the kernel more modularized and providing the selection of what modules of the kernel should be included in each part of the agent, thus, decreasing the weight of each part of the agent. In spite of it, the 40% of the necessary elements are common for all protocols. But it must be noted *libmodel-specific* library is very model dependant, as it is the description of the model itself, so its size is very variable (See Figure 6.5). In this case, tits size is 11700 KBs.

Table 6.1: Libraries and apps needed by TRILATERAL.

Protocol	**KB**	**CoAP**	**HTTP**	**SOAP**
app-reporting-client-rest	675.1		■	
app-reporting-client-soap	1600			■
app-reporting-server-rest	1300		■	
app-reporting-server-soap	1600			■
app-service-server-coap	9600	■		
app-service-server-rest	8200		■	
app-service-server-soap	9700			■
libenv.a	1200			■
libgsoap++.a	1600			■
libjsoncpp.a	2000		■	
libcoap.a	814.5	■		
libcbor.a	381.6	■		
libreporting-client-soap.a	2800			■
libreporting-server-soap.a	3000			■
libreporting-client-rest.a	2700		■	
libreporting-server-rest.a	2900		■	
libservice-server-rest.a	6900		■	
libservice-server-soap.a	7100			■
libservice-server-coap.a	6900	■		
libmodel-specific.a	11 700	□	□	□
libmodel-kernel.a	25 700	■	■	■
Total	96 671.2	43396.1	50375.1	54300
		45%	52%	56%

As TRILATERAL is a tool designed and developed in collaboration with another Ph.D. student, the evaluation and validation of the agents was not part of this thesis. Despite this, the evaluation and validation of it is presented in the following articles

[IUCMMU18, ICMU18b, IIL+19b].

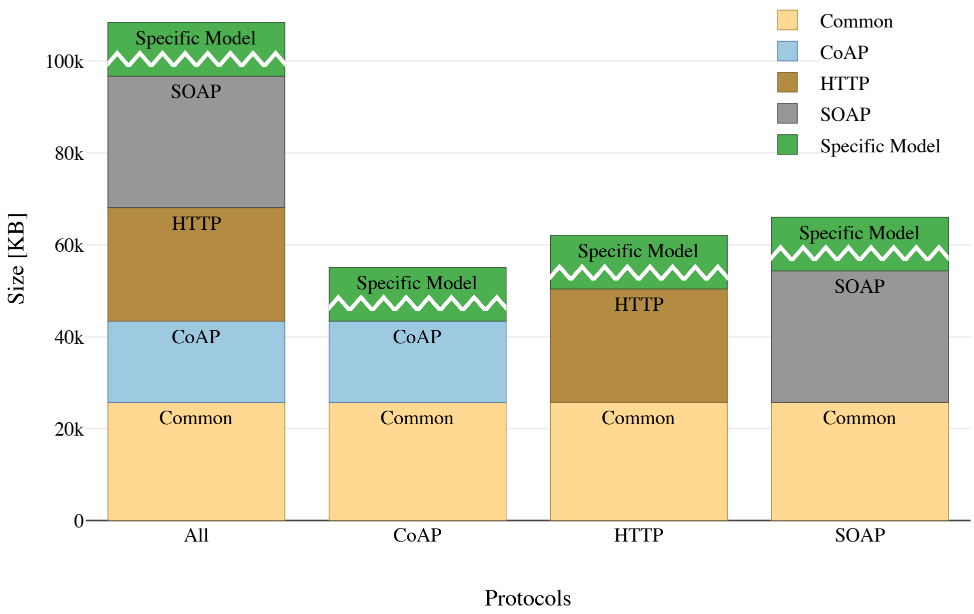


Figure 6.5: Different protocol sizes

In summary, TRILATERAL has allowed to improve the engineering process of ICPS agent development, not only reducing time and costs but also improving the validation and maintenance tasks.

After TRILATERAL was developed and validated with a CFT use case, it is noted that even if the use case is related to transport system, the proposed solution can be used in any domain (manufacturing, energy, etc.), since the Internet of Things (IoT) 61850 kernel code is useful outside the electrical substations. Finally, it can be seen how beneficial the paradigms System Product Line (SPL) and Domain Specific Language (DSL) can be. Because, even though many domains exist they share similar requirements and many commonalities exist between them. In addition, although the DSL development was complex, once it was well designed and developed, the configuration of an IoT communication protocol becomes much simpler, mainly due to the use of a visual environment.

6.1.3 Personalized Visualization & Evolution Detection System

Once it was verified that the *ICPSMonitoringProduct* was able to start monitoring devices through agents, the *personalized visualization & evolution detection system* was developed in order to prove: (I) if the user was able to create Graphical Elements (GEs) taking into account the information stored in the NoSQL database; (II) that the user was able to update the already created GEs without stopping the *ICPSMonitoringProduct*; (III) that the system was able to detect and classify the ICPS evolution; and (IV) if the system alerts correctly the ICPS evolution to the end user.

Different technologies were used for the *personalized visualization & evolution detection system* development as shown in Figure 6.1. It is a web application which was developed in SpringBoot using Java-programming language, i.e., server-side rendering architecture. Although the general logic was programmed in Java, other tools were used to achieve some functionalities. Note that these services are independent of the *personalized visualization & evolution detection system* logic. Depending on the customer's criteria, these services will be different. Thus, it is possible to generate a customizable system taking into account user requirements. In our case, the external tools used were ElasticSearch, Javers, and Infogram.

Once the data captured from the ICPS was stored in ElasticSearch in the JSON format by the *ICPSMonitoringProduct*, the *personalized visualization & evolution detection system* is able to detect if a new data model is introduced. Therefore, when a new data model exists, the data will be captured by SpringBoot as Java objects. Then our system will compare if any evolution has occurred using JAVERS, which provides facility to compare different versions of textual models [SC13]. Once the Diff Model is generated, the *personalized visualization & evolution detection system* evaluates which GEs are affected, and it generates the corresponding alerts for every GE. In addition to generating the corresponding alerts, the GE will also be updated, if needed. Infogram API is used for generating or updating the corresponding code of every GE. The connection to Infogram API is developed in Java-programming language, and then the necessary configuration parameters and the corresponding selection model are introduced. This way, Infogram returns a script, which is used by the *personalized visualization & evolution detection system* to display the GE to the user. The developed *personalized visualization & evolution detection system* project is about 241.664 bytes and is composed by 53 files.

In order to validate that the *personalized visualization & evolution detection system* was working correctly, a real dataset provided by ULMA was used.

The dataset corresponds to a small AW composed of 55 devices of five different types, which can be classified into five different areas. Due to confidentiality reasons, it is not possible neither to show data that each device is sending, nor real information

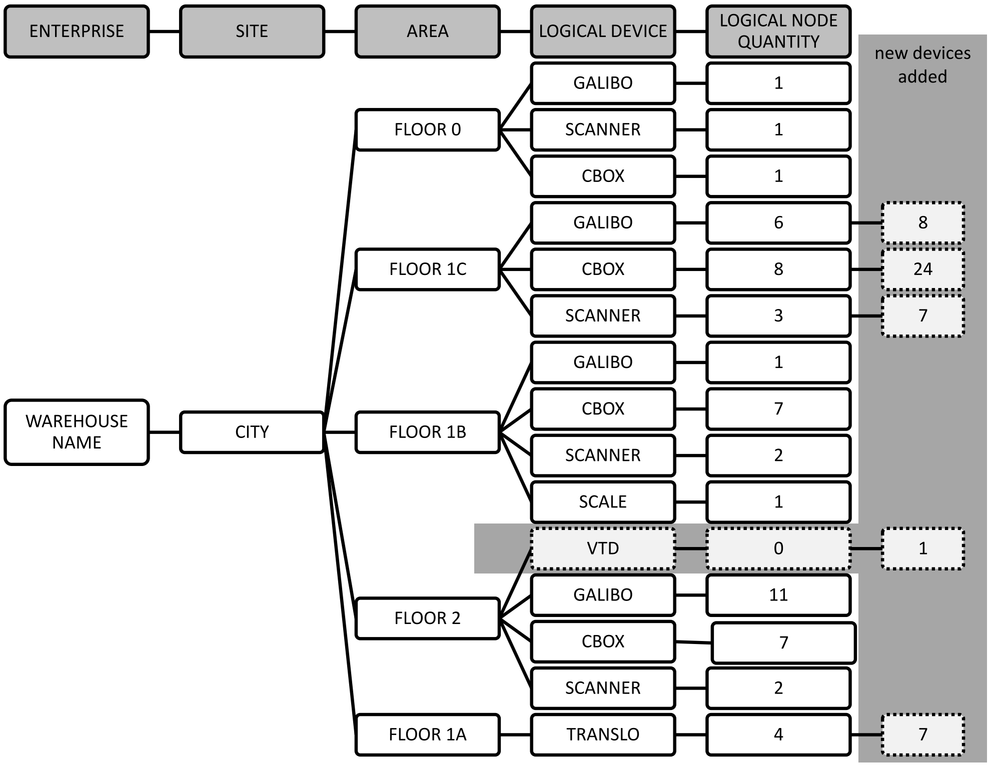


Figure 6.6: The Dataset Structure from the Real AW to validate the *Personalized Visualization & Evolution Detection System*

about the AW. Thus, Figure 6.6 only presents a layout that shows the structure of the storage facility without giving further details. As explained in previous chapters, our dataset is composed of seven different levels. With the fifth and sixth levels, i.e., the *LogicalDevice* and *LogicalNode*, the physical devices is defined in the dataset. As it is shown in Figure 6.6 in the penultimate column, the number of devices inside each *LogicalNode* is presented in order to highlight the number of devices and how the data captured is distributed. Note that each device has its data, i.e., data attribute with the corresponding data attribute value. For each timestamp, one value for each data is captured.

Even though the first dataset captured was composed of 55 devices, the AW has evolved over the time. ULMA's customers' needs have grown as their productivity has grown and the capability of the AW they had needs to be extended. This resulted in a growth of 68% of devices in the AW, i.e., from 56 devices to 81. Also, some new devices were introduced to the AW (e.g., VTD in floor2) as shown in Figure 6.7. The new AW has six types of devices as compared to five in the previous AW.

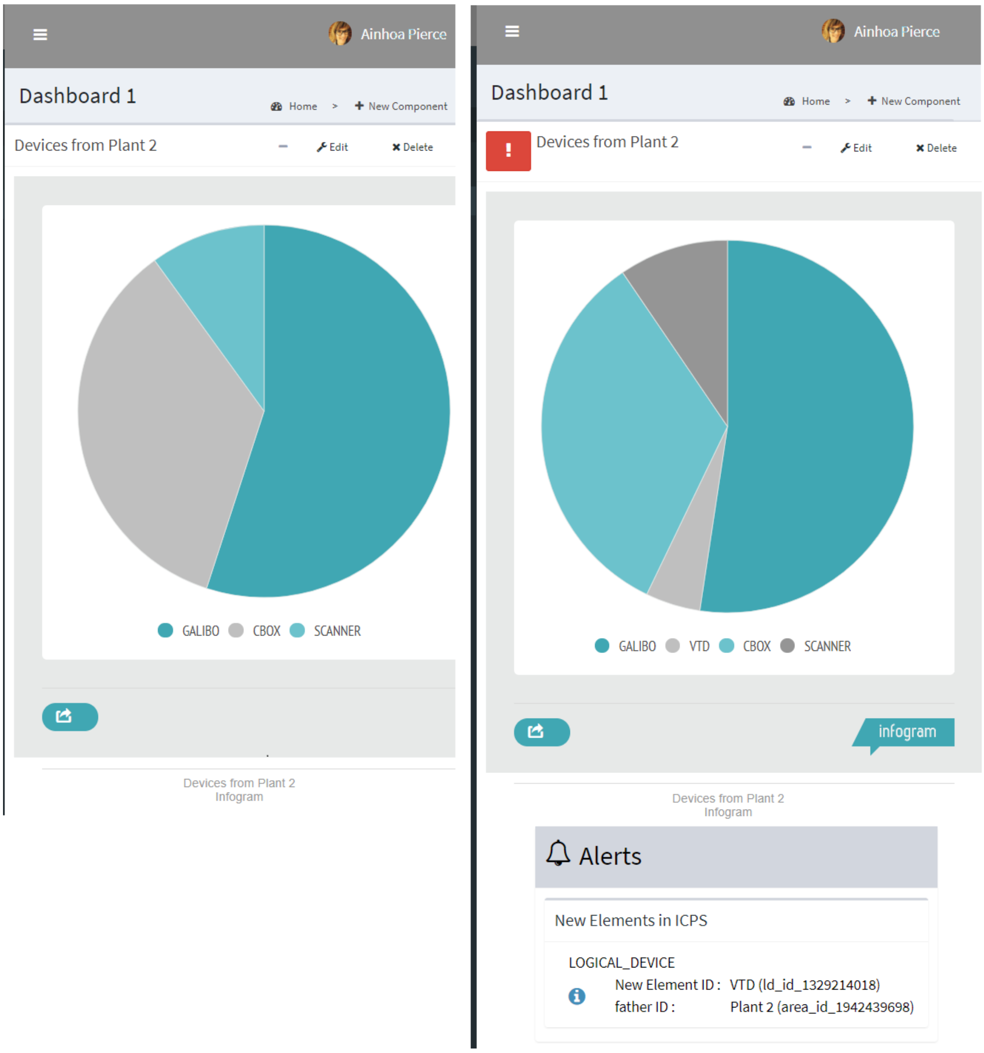


Figure 6.7: Alerts when ICPS evolves. Left) before evolution. Right) after evolution (new *LogicalDevice* is introduced)

Based on this dataset, the developed *personalized visualization & evolution detection system* was validated with a real dataset but not in production. First six visualizations were created using the *personalized visualization & evolution detection system*. Once these ones were created, some of them were updated in order to prove that it was possible for the user to update already created GEs. Afterward, the ICPS evolution was simulated manually by introducing the dataset in the database. Thus, once the dataset from the ICPS is captured by the system this one checks whether the ICPS has evolved. If so, it automatically creates the Diff Model using JAVERS and it stores in ElasticSearch. Because of this, both, the reception of data (every timestamp) and the ICPS evolution were tested. Since if any GE needs to be adapted the *personalized visualization & evolution detection system* adapts the dashboard to the changes in runtime, in addition to informing users about the changes. Figure 6.7 shows how the visualization evolves over the time. Note that a new device has been introduced (e.g., VTD in floor2, see Figure 6.6), which is a relevant information for the user, even if it is new, i.e., it did not exist before, the *personalized visualization & evolution detection system* can notify the user about the evolution and start capturing all the information about that new device. Also, the *personalized visualization & evolution detection system* gives the user the possibility to create new visualizations, if required. Due to confidentiality reasons, Figure 6.6 and Figure 6.7, only show the number of devices, but actually, it is possible to visualize all the information coming from the ICPS even if a new device is introduced.

Based on the validation of the *personalized visualization & evolution detection system* with a real dataset, it is concluded that the application of this system is useful as long as it meets the specified requirements, i.e., that the data is saved on the basis of the created Data MetaModel. Thus, with the developed system it is possible to inform users about all changes and adapt visualizations considering the ICPS evolution. New visualizations can be obtained more easily without requiring coding by a software developer.

6.2 Evaluation

In this Section, firstly the Return Of Investment of the *modular monitoring system* and the *personalized visualization & evolution detection system* considering technical managers opinion is presented. As second point, a Questionnaire-Based Survery (QBS) is performed in order to analyze the interest of the both systems in industry. Finally, the section is finished with a scalability test where the usability of the ICPS *evolution detection system* is evaluated in order to be used in different domains and with different ICPS sizes.

6.2.1 Return Of Investment

In the next sections the opinion of a technical manager about the developed solution in terms on investment return is analyzed. First the *modular monitoring system* is presented and then the *personalized visualization & evolution detection system*.

Modular Monitoring System

With the objective of evaluating the resource-effectiveness (e.g., cost, time) of the developed prototype, i.e., the concrete *ICPSMonitoringProduct* for AW domain, collect customer feedback, and with the final goal of obtaining the technical managers' opinion of the *modular monitoring system*, a QBS was filled by one of the technical managers of ULMA Handling Systems (see Section C.2). The prototype of *ICPSMonitoringProduct* was supervised by the technical manager, that is why, it was interesting to asked its opinion about the developed prototype.

Taking into account that different ICPSs of the same domain can use the same type of devices, they also use the same agents for monitoring those devices. That is why, knowing that commonality exists, the technical manager of ULMA Handling System was asked about how many agents are usually reused from one ICPS to another. According to him, 50% to 99% of the agents are reused between different ICPSs in the same domain, i.e., AW. Therefore, it can concluded that a high percentage of devices are reused. Thus, when a new device is introduced in the ICPS, the likelihood of reusing the code of the agent is high.

In addition, the manager was asked about the development and commissioning time. The development time refers to the time that a developer needs to develop the corresponding agents to be used in an ICPS, either for a new installation (a new ICPS needs to be monitored), or for introducing a new device within the ICPS (the code of the agent of the new device). However, the commissioning time refers to the time needed to get these agents up and running to start monitoring data from the devices located in an ICPS. The survey results are shown in Figure 6.8.

Concerning a new installation, the average time required is the same in both development and commissioning time (between 6 and 10 days). However, when an ICPS evolves (e.g., a new device is introduced), the average development and commissioning time is about 20 days (between 10 and 30 days). Note, that the insertion of a new device implies the generation of a new agent for monitoring the introduced device. Hence, the effort needed for introducing a new device is higher than the effort and time needed to start monitoring a new ICPS in the same domain, e.g., AW domain.

Considering technical manager's opinion, in AW domain, the 50% to 90% of the agents are reused, thus, the time required for the introduction of new devices may

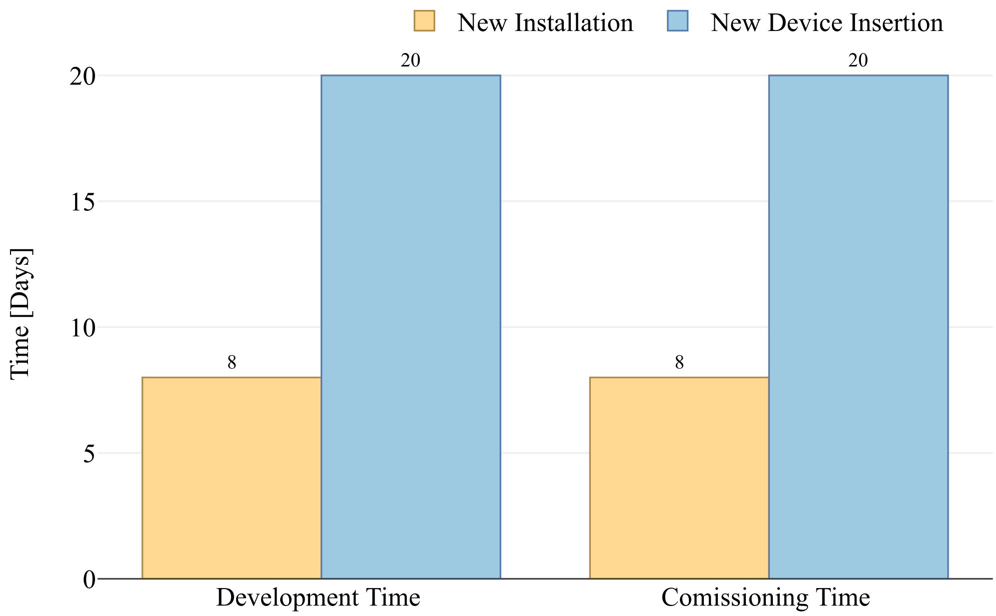


Figure 6.8: Development and Commissioning time currently required for 1) a new ICPS installation or 2) the introduction of a new device into ICPS according to the technical manager in AW domain

be considered somewhat long compared with the development or commissioning time needed for a new installation. As the development of a new agent is necessary when a new device is introduced, in case that device is never used in that domain. Consequently, if the device exists and it is used to monitor another ICPS in the same domain, the agent associated to that device should not be created, since it already exists in the common repository, i.e., the agent exists in the common repository of ULMA Handling Systems.

Although there is no quantitative information on the time of development of the system or the flexibility given by the *modular monitoring system*, the feeling and opinion of the customer is positive. Since it is a configurable system that can be adapted to the needs of customers, it reduces the start time of monitoring an ICPS, avoiding having to develop a monitoring system for each end customer from scratch. The assessment is positive whether in initiating the monitoring of a new ICPS or in managing evolution itself. In the future it would be interesting get more quantitative information. In this way it will be possible to evaluate the improvement that the *modular monitoring system* has provided in comparison with what they had previously.

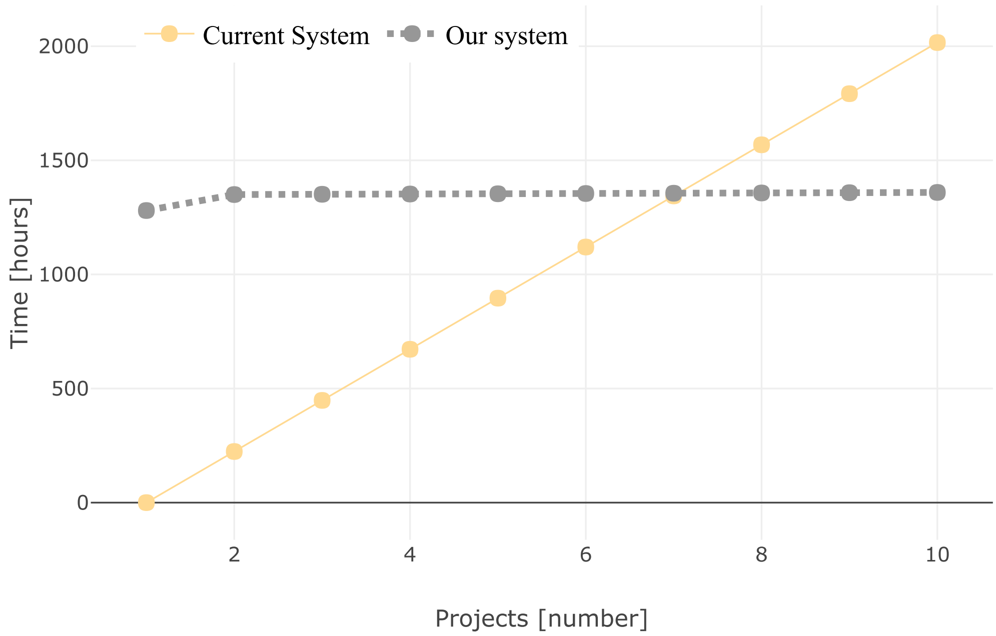
Personalized Visualization & Evolution Detection System

As well as the opinion about the *modular monitoring system* was analyzed, it was interesting to analyze the opinion about the *personalized visualization & evolution detection system* in terms of usability.

The use of model-based solution permits faster generation of a concrete monitoring software system in addition to visualizations. ULMA estimates such effort reduction based on its current practice as discussed below.

Thanks to the personal interviews with the technical manager it is estimated that more or less in current practice, on an average, 32 hours are needed to develop a new dashboard comprising of, on average, six GEs. Such a service is often outsourced. In contrast, it is estimated that the *personalized visualization & evolution detection system*, needs, on average, 10 minutes to generate a graphical component; therefore, taking, on average, one hour for generating a dashboard. Thus, the current practice takes roughly 31 hours, more than our system. Consequently, a reduction of effort by 97% regarding the development of each of the dashboards can be expected.

However, it shall be considered the design and development time of *personalized visualization & evolution detection system*, i.e., about eight-person months. However, this is a just one-time effort. Although *personalized visualization & evolution detection system* takes more time at the beginning, when considering its development cost, it is profitable in the future as shown in Figure 6.9. In Figure 6.9 it can be appreciated the comparison between our system and the current version (i.e., the one different industrial domains provides to its customers) where how many hours are needed to generate the visualization corresponding to each project is observed. A project is a visualization with the necessary GE's that belongs to an ICPS. Based on data provided by the technical manager and considering the development cost of the *personalized visualization & evolution detection system*, it will pay off from the seventh project. Note that the time needed for each new visualization or modification inside a dashboard is shorter with the *personalized visualization & evolution detection system*, as in some cases the code is automatically generated. Also, with the presented system, a user who creates GEs does not need to be an expert. Nevertheless, Figure 6.9 shows that the benefits of the *personalized visualization & evolution detection system* will grow along the time. Note that the *personalized visualization & evolution detection system* has not been integrated into the production environment yet since it is a prototype. Additionally, if the necessary GE meta models are not created, these ones also need



Projects	0	1	2	3	4	5	6	7	8	9
Current System [hours]	0	224	448	672	896	1120	1344	1568	1792	2016
Our System [hours]	1280	1350	1351.17	1352.33	1353.5	1354.67	1355.83	1357	1358.17	1359.33

Figure 6.9: Comparing the *personalized visualization & evolution detection system* with the Current Practise

to be developed by the developer, hence, the *personalized visualization & evolution detection system* development may also increase over time. Thus, although in the graph it is taken into account that for all projects the same development time is necessary, it must be remarked that it will depend on the project itself. That's why, in the future, it would be interesting to convert this prototype into a real product and hence, analyzing and evaluating the real return of investment since the presented results are an estimation.

6.2.2 Survey about proposed solution

After analyzing the return of investment, the interest of managers, developers, users, and designers on the created prototypes was analyzed since introducing a new technology might significantly impact the current practice in the industry, therefore, hinders its adaptation. Thus, a QBS was conducted and sent it to the industry to solicit views of experts (See Appendix C) in order to know the real interest of people who are working in those domains.

The questionnaire was sent to people working in AW and Press Line (PL) domains. Collecting information from multiple domains helped to validate if the solution can be applied to multiple domains. Based on the responses, the results were analyzed to validate the usability and interest of the developed prototype.

The participants of the questionnaire were professionals with different backgrounds, i.e., managers, developers, users, and designers working in the *monitoring systems* and data *visualization systems* domains. Managers and supervisors have sufficient knowledge about the ICPSs in their respective domains and are managing the technical staff. Users are the participants who are the potential users of the visualization system for monitoring ICPSs. Designers and developers are responsible of developing and designing visualization and monitoring systems. The requirements for different roles that would potentially lead to ease of adoption would be validated. All the questions in the questionnaire were on a five-point Likert Scale to solicit responses.

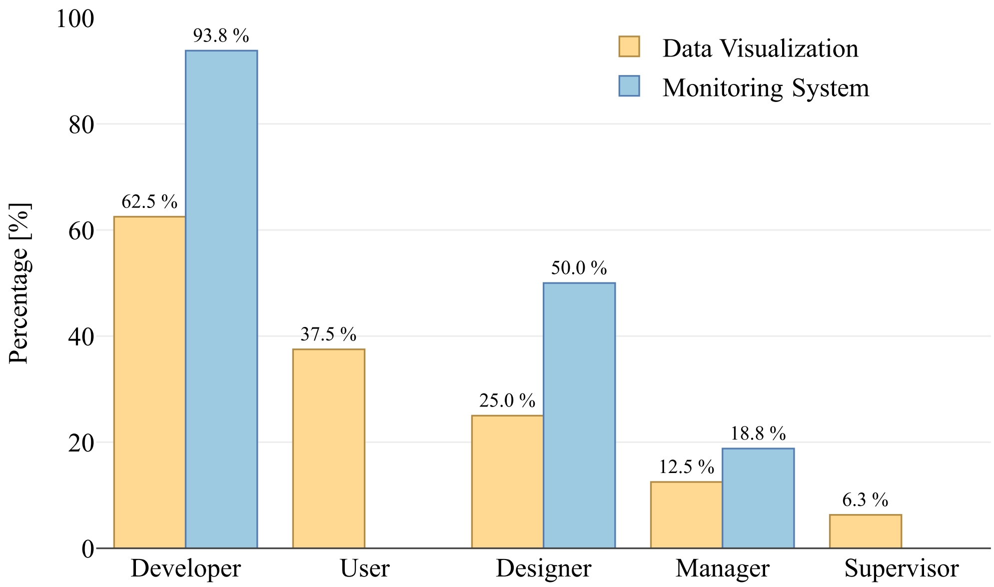


Figure 6.10: Participants' Roles

The department dedicated to the monitoring, analysis, and visualization of data in the hosting research center (IKERLAN) has 38 people, and 18 of them are dedicated to working with different industrial environments for data monitoring, analysis, and visualization. Also, IKERLAN made contact with two external people (experts in industrial domains) who are in charge of carrying out this type of project together with

IKERLAN. Thus, the questionnaire was sent to 20 participants, and 16 responded, i.e., an 80.0% response rate. Note that even the sample size is small, the selected participants represent relevant roles in different industrial domains, i.e., AWs and PL. The participants were also involved in monitoring ICPSs, in addition to having general domain knowledge of their respective ICPSs.

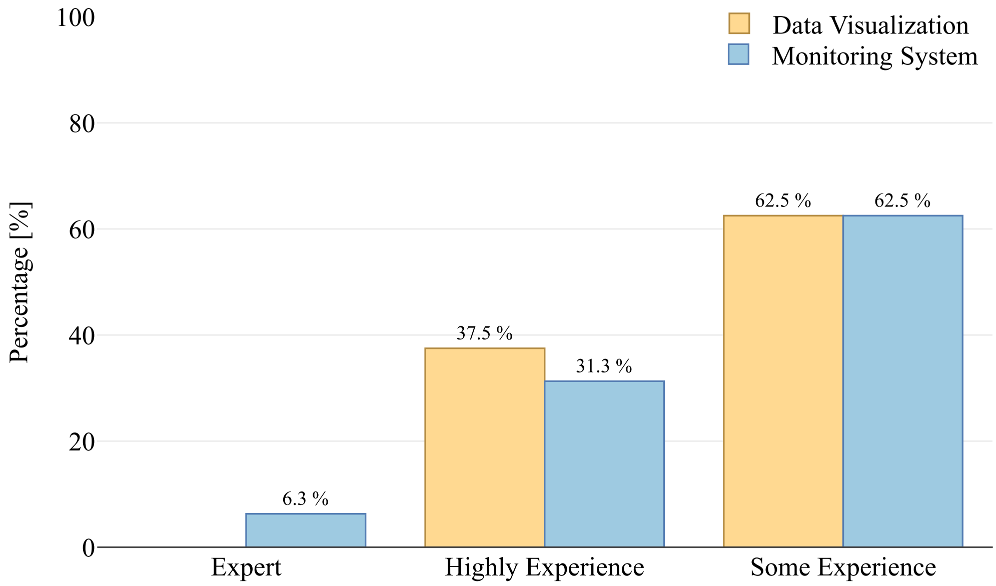


Figure 6.11: Knowledge about Data Visualization and Monitoring System

Figure 6.10 shows the overall distribution of the roles of the participants for data visualization and monitoring system. When analyzing the monitoring system results, 93.8% of participants considered themselves developers. Out of these (see Figure. 6.11), 62.5% considered themselves to have Some Experience, 31.3% considered them as Highly Experience, and 6.3% considered themselves as Experts. For data visualization, the roles are more diverse. Even if most of the participants considered themselves as developers (62.5%), no one considered her/himself an expert. The majority of the participants (62.5%) considered that they have some data visualization experience.

The level of importance of the following items were validated: (I) visualizing data; (II) collaborating with users for configuring the GEs of a User Interface (UI) and the benefits of doing so; and (III) the importance and the benefits of detecting ICPS evolution.

The majority of the participants (75.0%) considered that it is important to display monitoring data of ICPSs at runtime. The remaining 25.0% believed that it is interest-

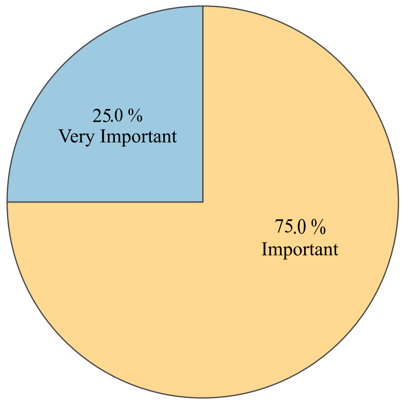


Figure 6.12: Importance of Visualizing Data

ing to them (see Figure 6.12). Overall, the participants believed that giving a chance for users to configure GEs at runtime is beneficial. The 81.3% of the participants believed that an overview of the operation of ICPS at runtime via effective data visualization is important. The 75.0% of the participants thought that detecting anomalies would be faster with a tailored GE. The 93.8% of the participants considered that a tailored GE would help to more efficiently propose improvements to ICPSs (see Figure 6.13). Based on the results, it can be concluded that visualizing data of ICPSs is very interesting for people who work with monitoring systems. According to the participants, due to the visualization and user collaboration, the proposed ICPS improvements can be provided more efficiently and detecting anomalies can be quicker.

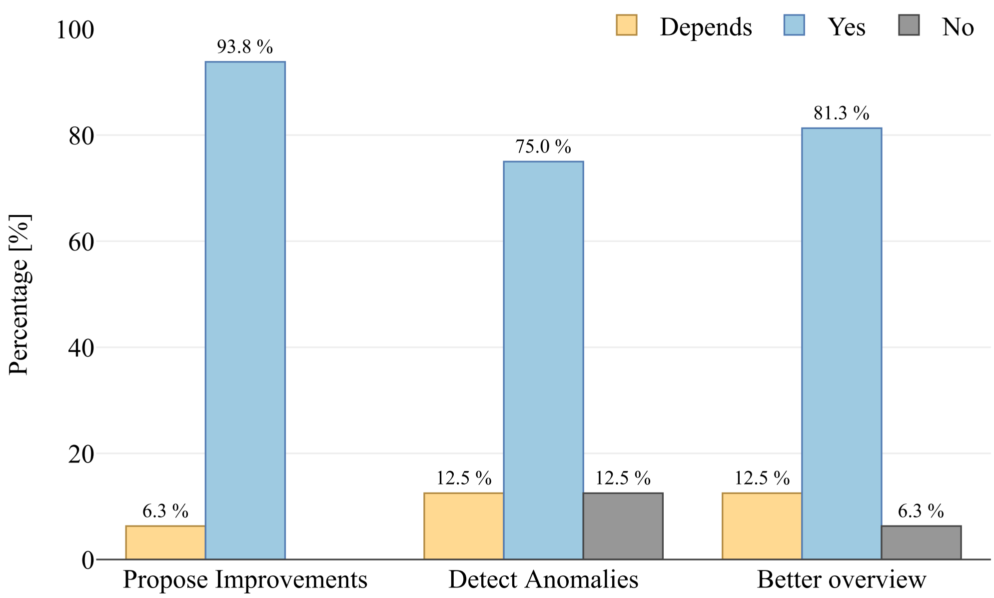


Figure 6.13: Challenges displaying data at runtime

In terms of detecting ICPS evolution as shown in Figure 6.14, the 100.0% believed that detecting ICPS evolution in automatic way is very interesting (87.5%) or interesting (12.5%). In addition, the 68.8% thinks that detecting the evolution at runtime is very interesting and the 25.0% thinks is interesting.

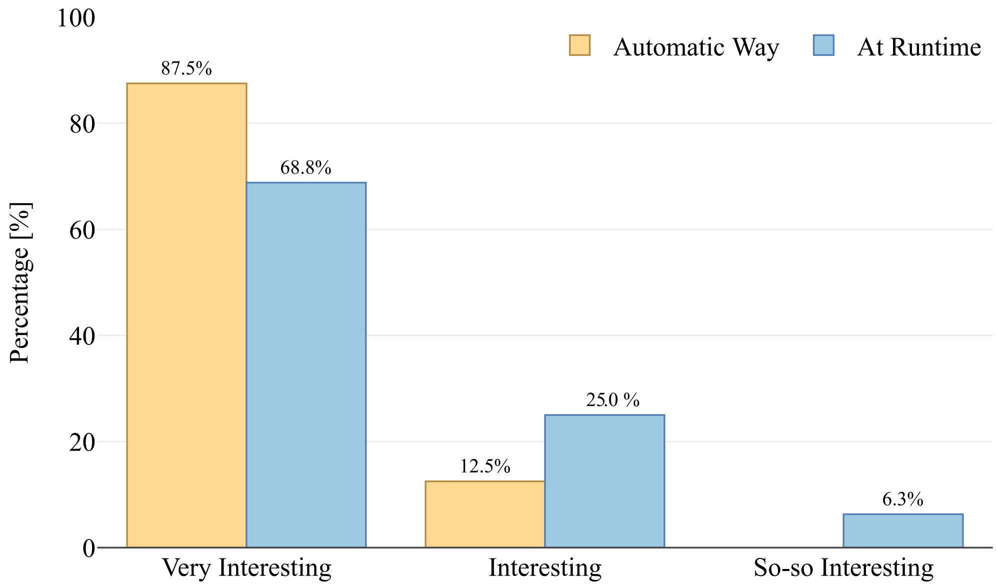


Figure 6.14: Detecting ICPS evolution at runtime or automatically

Based on the results, it can be concluded that an efficient system able to visualize ICPS is performed since visualizing data of ICPSs is very interesting for people who work with monitoring systems. According to the participants, due to the visualization and user collaboration, the proposed ICPS improvements can be provided more efficiently and detecting anomalies can be quicker. Additionally, detecting evolution at runtime and automatically in order to be aware of what is happening in the ICPS is an interesting topic.

Thus, considering the results, both prototypes are interesting for user, since with the modular monitoring system is possible to manage ICPS evolution and with the *personalized visualization & evolution detection system* is possible to visualize that information even if an evolution occurs.

Threats to Validity

Regarding internal validity, the participants may not have comparable levels of knowledge or experience, and this may affect the conclusions. The participants were selected based on their role, knowledge, and experience to have a balanced group of participants; however, many of them did not consider themselves as experts. As for external validity, the limited number of participants is a threat. In this case, a sample size of 16 was used and note that it is very challenging to arrange a large number of participants for such surveys in the industry. It was also realized that the answers are unbalanced since most of the participants consider themselves as developers. This affects the generalization of the results obtained in the study. Finally, regarding construct validity, more direct contact, e.g., personal interviews, with different industrial environments will be realized in addition to get more information about each domain and realize more specific and personalized questions. In this way, further information about the identified requirements with the objective to give them a better solution would be identified.

6.2.3 Scalability test

After implementing and validating the system with a real data set, it is noted that despite ICPSs evolution, different ICPS Data Model sizes exist, e.g., a press machine can be composed of 50 or more than 1000 devices, this being one of the machines within the PL. In addition, a customer can also be interested on monitoring multiple PLs, thus the size of the environment can be huge. Thus, in order to ensure the system will be applicable in a real scenario, in particular the *ICPS evaluation detection subsystem*, a scalability test with different ICPS sizes is provided.

The data to perform the scalability test were created in a random way, since there was no access to real data. Thus, considering that the *modular monitoring system* generates every Data Model that conforms to the Data MetaModel, these Data Models were randomly simulated. In this way, it can also be concluded that the presented subsystem is applicable in other domains that use the Data MetaModel as a basis, as long as the scenario to be monitored complies with the Data MetaModel characteristics presented in Chapter 4.

All the experiments have been performed on a laptop with a CPU Intel (R) Core(TM) i7-4600U CPI @2.10GHz CPU. In addition, the computer in use has 16GiB memory with a 64bit operating system (Linux) and a 500GB disk (HDD). The correctness of the Diff Models has been tested manually with a smaller Data Model. Java Microbenchmark Harness (JMH)[4] was used to execute an accurate mi-

[4]https://www.baeldung.com/java-microbenchmark-harness

crobenchmark in an automatic way, i.e., measure the average time needed to run and the confidence interval of the average.

In order to get reliable numbers, each query was processed 200 times for each evaluation case and Java Virtual Machine has been restarted for each execution for each test.

Considering that changes can occur in any of the Data Model levels but mostly the evaluation occurs at the devices (i.e., Logical Node), in the evaluation, changes in this level were simulated. For the evaluation, as mentioned above the Data Models were randomly generated with 50, 100, 500, 1000, 5000, and 10000 devices. Each Data Model has been cloned and changes performed: addition, removal, modification and random changes (addition, modification, removal). Finally, different percentages of changes were established (from 20% to 100%). Note that 100% means an addition of 10000 devices in the largest model or a modification of the devices. In the case of removal, the removal of 100% was skipped, as it would result in an invalid Data Model. The scalability test results are reflected in Appendix D.

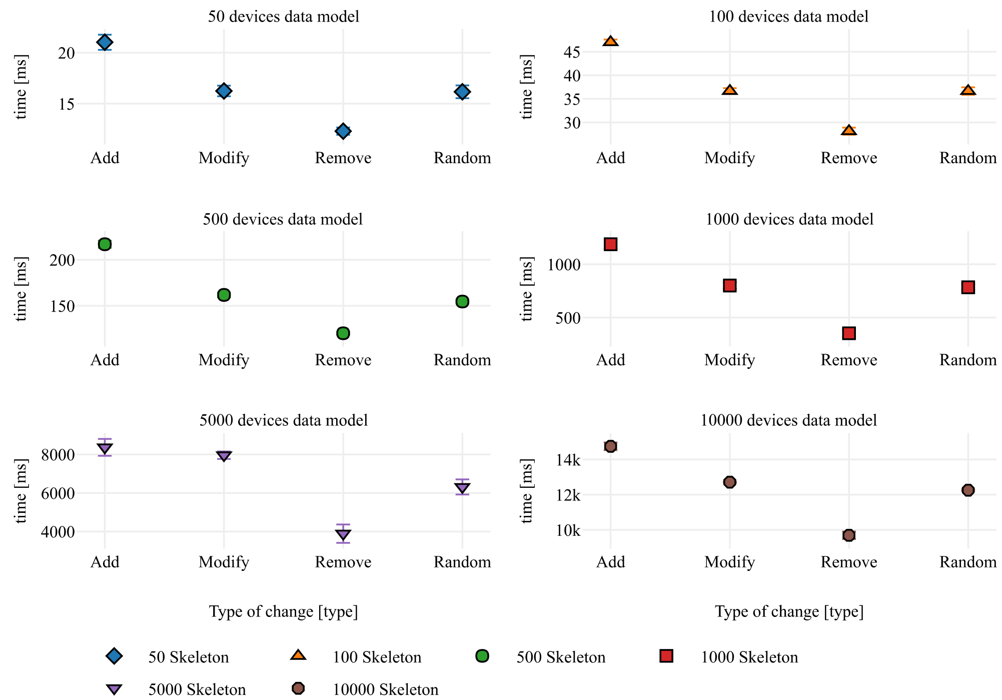
In this evaluation, the following question were addressed: ***which is the performance of the evolution detection and classification subsystem?*** To do so, three different configuration factors (F) that may impact were distinguished:

- **F1 → Type of change:** How the type of change (addition, modification or removal) impacts on the performance (execution time) was measured.

- **F2 → Percentage of devices changed:** How the number of changed devices impacts on the performance (execution time) was measured.

- **F3 → Size of the Data Model:** How the increasing size of the Data Model impacts on the performance (execution time) was measured.

Considering all the results obtained, the following sections discuss the factors F1, F2 and F3 as well as a joint analysis of all. Each factor has been evaluated with the following quantitative metrics: AVG: Average Execution Time in milliseconds (ms) and CI: Confidence Interval (ms).



Size	Insert (ms)	Remove (ms)	Modify (ms)	Random (ms)
50	21.0176 ± 0.76	12.30825 ± 0.3645	16.2452 ± 0.5166	16.666 ± 0.6328
100	47.031 ± 0.5842	28.1455 ± 0.7605	36.6702 ± 0.616	36.6632 ± 0.801
500	216.7906 ± 3.9016	120.181 ± 1.609	161.8192 ± 2.5052	134.6054 ± 1.6938
1000	1187.855 ± 45.1296	353.148 ± 35.8075	80.7182 ± 31.8412	787.4724 ± 26.7418
5000	8366.062 ± 439.4848	3889.366 ± 475.1935	7957.895 ± 195.502	6312.037 ± 391.2232
10000	14750.11 ± 217.8432	9688.494 ± 231.494	1714.11 ± 34.9484	12257.34 ± 45.1536

Figure 6.15: Execution time average in different ICPS Data Model sizes

F1 → Type of change

In Figure 6.15 shows the different ICPS Data Models separated by the different changes, i.e., inserted, removed, modified. The type of changes seems to affect our system. That is, the detection of adding a device is not the same as detecting that a device is removed or modified. Taking as an example the Data Model of 50 devices, adding devices average is 21.0176ms ± 0.738ms, removing devices is 12.30825ms ± 0.3645ms and modifying devices 16.2452ms ± 0.5166ms. In the case where all kind of changes are made in the same Data Model (insert, remove, modify), the time needed is 16.1666ms ± 0.6328ms. The difference between removing and inserting devices (maximum and minimum execution time) is about 71% for this Data Model.

Considering the differences between the highest and lowest execution time of all Data Models (see Table 6.2) it can be concluded that it does not have a relation with the size of the Data Model. But in all the cases the difference between the maximum and the minimum is of 50% up to 220%.

Table 6.2: Max and Min execution time difference percentage

	50	100	500	1000	5000	10000
Difference between insert and remove	71%	67%	80%	224%	115%	52%

The type of change to detect and classify the evaluation affects the result. Thus, detecting removals is less expensive than detecting insertions in a Data Model. In the same way, the execution time of modifying devices is between adding and deleting. In the case of random changes, the maximum (insert) and minimum (remove) time are compensated and therefore, the average time achieved is more or less in the middle.

F2 $\rightarrow$ Percentage of changes

This factor evaluates if the percentage change affects the execution time, i.e., if with the same Data Model (e.g. 1000 devices), changing 20% (e.g., 200) or 100% (e.g., 1000) of devices influences on the time required (execution time) for detecting changes. Results are shown in Figure 6.16.

Considering the different changes that can occur it can be observed that:

- **Inserting:** The higher the percentage of change, the higher the execution time. In the smallest Data Model (50 devices) the execution time has an increase of 41.89%. With a bigger Data Model (e.g., 10000) the difference is 46.49%. Considering all the results (see Table 6.3), the difference between the minimum and maximum execution time when the percentage of change changes is more or less between 40% and 60%. Therefore as it is reflected in Figure 6.16, the growth of time is linear to the percentage of change.
- **Removing:** The lower the percentage of change made, the longer the execution time. In addition, the time decreases linearly. The execution time required for removing 20% in a 50 device Data Model is 14.462ms but for

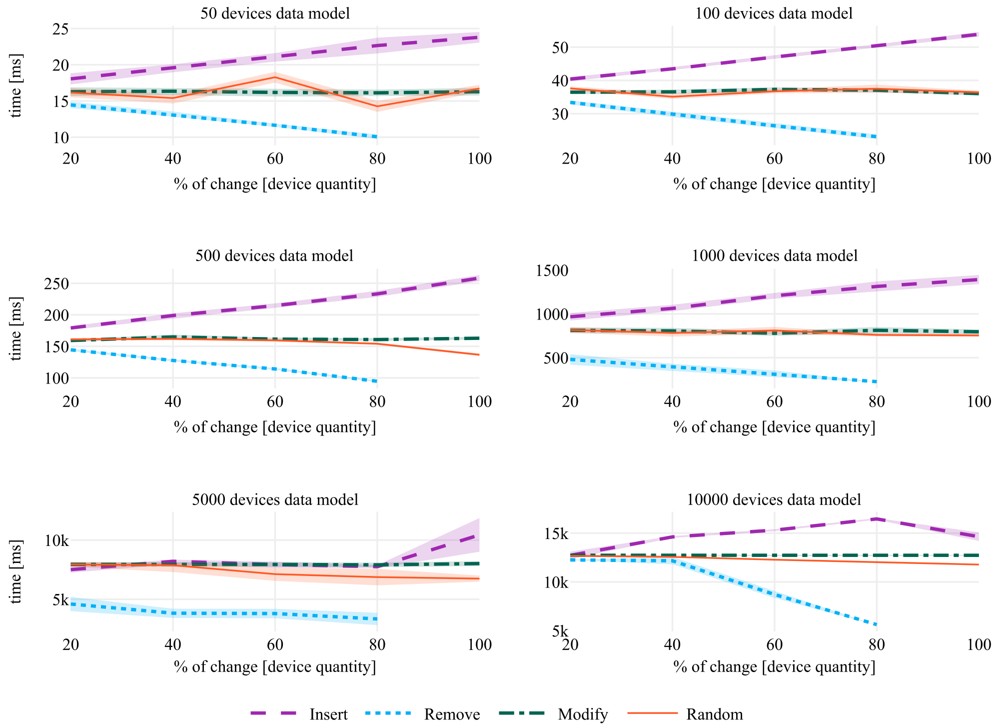


Figure 6.16: Increase the percentage of changes in different ICPS Data Model sizes

80% is 10.06ms. A difference of 52.86% exists, reaching 143.98% in the case of 1000 devices.

- **Modifying:** Increasing the percentage of change does not have a negative impact on time. The major percentage of change occurs with 1000 Data Model size, i.e., 0.013%. That is why it can be concluded that the trend is constant.

- **Random:** This scenario is similar to modifying devices, i.e., increasing the percentage of change does not have a negative impact on the time, making the trend rather constant. Besides, this is a case that highly depends on the changes that have taken place.

Therefore, in a common scenario where it is not possible to control what is going on, it is assumed that the percentage of change does not have a negative impact on the execution time needed. The times of adding and removing are compensated for each other leaving a rather stable time when making random changes.

Table 6.3: Difference between the minimum (remove) and maximum (insert) execution time

	50	**100**	**500**	**1000**	**5000**	**10000**
% of ADD	41.89%	37.27%	49.21%	56.20%	63.51%	46.49%
% of MODIFY	7.19%	7.02%	7.40%	12.57%	6.44%	0.60%
% of REMOVE	52.86%	52.00%	56.70%	143.96%	84.63%	120.71%
% of RANDOM	40.92%	12.08%	7.71%	15.49%	134.10%	8.19%

F3 → Size of the Data Model

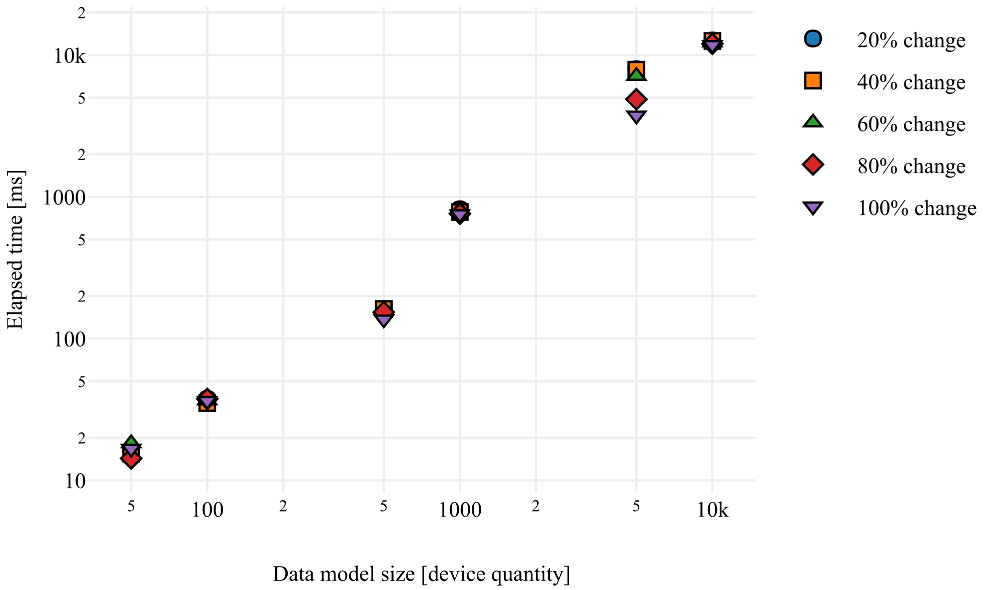


Figure 6.17: Time need for detection versus ICPS Data Model sizes (the axis are in logarithmic scale)

In order to see if the ICPS Data Model size affects the execution time needed, the values of the random test case were extracted (see Appendix D, Table D.7). As concluded in factor F2, in a real scenario it is not possible to know about the change that is going to happen, and it has also been shown that time is more or less constant in terms of the number of changes that have occurred. In this manner, all the cases without focusing on a single change were contemplated. As it is shown in Figure 6.17, the larger the ICPS Data Model, the longer the execution time is. For detecting 100% of changes in a 1000 device Data Model, i.e., 1000 changes, our subsystem needs an average of 754.174±736.894ms, in contrast in 5000 device Data Model detecting 20% of changes, i.e., the same quantity of changes (1000), the time needed is bigger

(7948.296±7740.268ms). For detecting the same quantity of changes, a 546% more time is needed, i.e., equivalent to 6.6% of seconds. Looking at the graph, it is noted that this is not an isolated case, it is something that occurs if different Data Model sizes are compared. That means, that the Data Model size, i.e., the input, impacts on the output. Since, the larger the size of the Data Model, the longer the time needed to calculate the differences even though the number of changes is the same.

In order to see the trend that our system has, the average needed for each ICPS Data Model size (considering random changes results) is calculated. Analyzing the results it is observed that they tend in a potential way, which can be represented as follows: $y = cx^a$.

Thanks to a linear regression, i.e., a mathematical model used to approximate the relationship of dependence between a dependent variable (time) and the independent variables (quantity of devices), the $a = 1.28852412$ and $c = 0.08764925519477608$ values are obtained.

Thus, the equation $f(x) = 0.08764925519477608x^{1.28852412}$ is obtained, which is represented in Figure 6.18 and shows the relation between the ICPS Data Model size and the execution time.

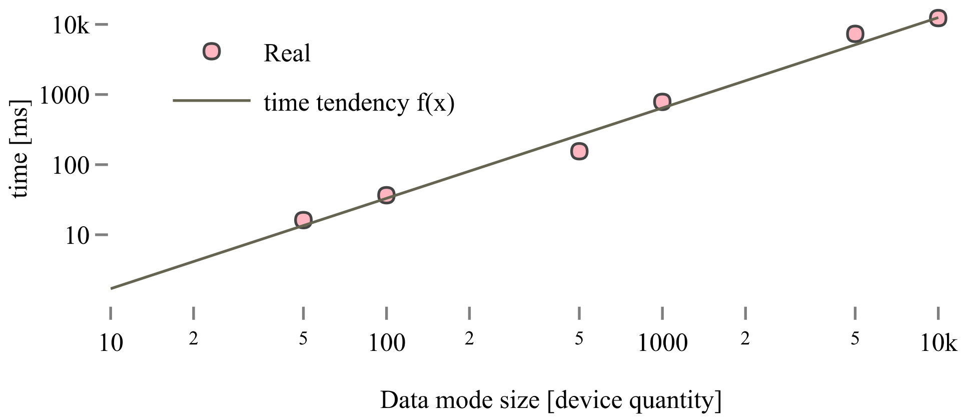


Figure 6.18: Time growth trend when increases the number of devices (the axis are in logarithmic scale)

Henceforth, using this solution with a high quantity of devices could be a problem, since if our subsystem needs to be used in an ICPS with many devices, it is necessary to give a solution to the scalability problem found in order to reduce the execution time. If a fast response is needed when the number of devices is high, this module would not be able to give a fast enough response to the user. For example, in a small (50

Data Model size) scenario, 0.016±0.0006s are needed, but a bigger one (5000 Data Model size) it needs 6.312±0.3912s. Usually, the latency of an industrial monitoring system is about 2000ms. That is why our system is not profitable enough in real time big Data Model scenarios.

Factor Analysis (F1, F2, and F3)

Considering all factors it is noted that, with small Data Models, the average time needed for communicating alerts is small. In the same manner, taking into account the Figure 6.18, it is noted that the behavior of the small Data Models is smoother than that of the larger ones as the trends are clearer. Thus, currently the subsystem is able to give response to small ICPSs, as long as it meets the customer's requirements, i.e., the latency is adequate.

In industrial scenarios in which monitoring has a latency that does not support the presented solution, thus, it is proposed to split the Data Model file into different files, making the comparison in each of the files. The following chart shows (Figure 6.19) the relation between the file size and the execution time where it is shown that the smaller the file the smaller the execution time.

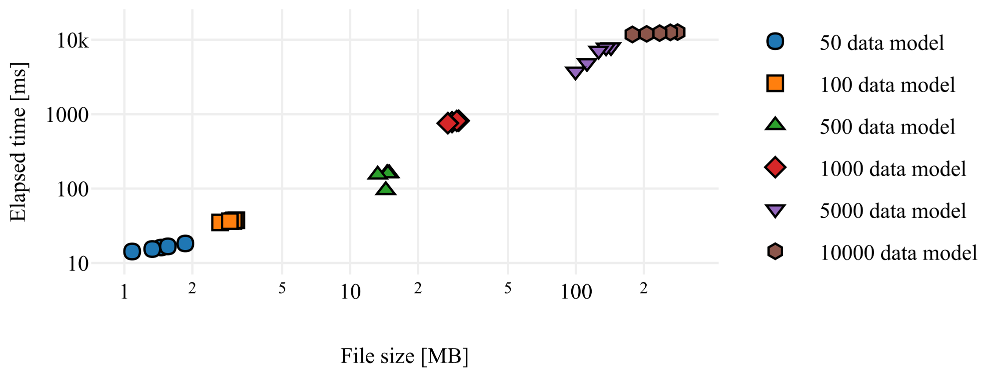


Figure 6.19: The relation between different Data Model file sizes and the elapsed execution time (the axis are in logarithmic scale)

Taking into account the results of Figure 6.19, it is proposed to divide the Data Models into sub-Data Models. Because there are no data dependencies between the tasks to be parallelized and hence, it can be used the parallel computing theory to decrease the response time. Dividing the model each sub-Data Model is smaller and thus, the individual execution time needed will be also smaller. In the same way, several sub-Data Models would be executed at the same time, so the total execution time will be reduced. But it is necessary to consider that a Deviation Time (det) exist, since time is needed to split the Data Model in sub-Data Models and then the result needs to be joined, i.e., the different sub-Diff Models need to be converted into a unique Diff Model, to finally transfer the information to the user. In Figure 6.20 is shown an activity diagram where using the parallel computing theory, the execution time can be reduced in order to give a response to the problem found.

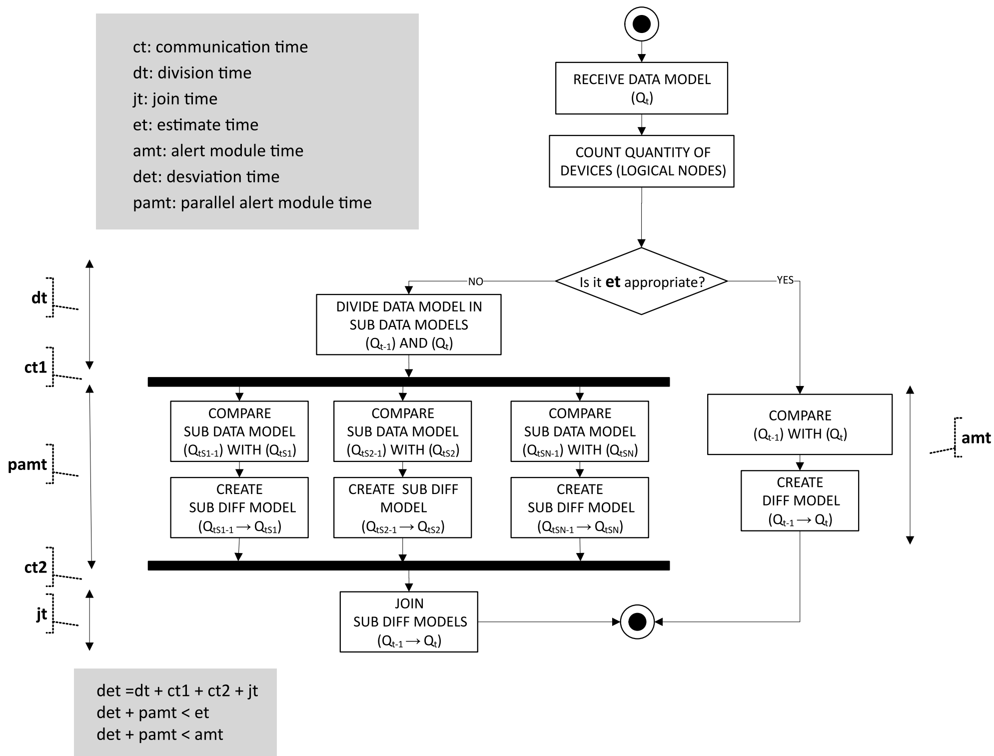


Figure 6.20: Diagram of activity in which it is estimated to reduce the result time of the alert module.

Note that for dividing the Data Model into sub-Data Models, is necessary to calculate the optimal division value, i.e., the value that instructs in how many sub-Data Models should be separated the model. To do so, the next points need to considered: (I) the time tendency formula (formula from Figure 6.18) and (II) the deviation time that the proposal will have (Figure 6.20). Thus, by developing this improvement and performing a scalability test, it will be possible to calculate the deviation time. Once the deviation time is calculated, the corresponding function to obtain the optimal value of the division can be calculated.

Making use of the parallel computing theory, is possible to divide the Data Model in sub-Data Models and calculate the optimal division value. Hence, it will be possible to genereate a system able to divide the Data Model in an efficient manner with the aim of parallelizing the comparison and thus, reducing time. Furthermore, in Big Data scenarios, thanks to the Map & Reduce programming models, there would be no problem in parallelization. Thanks to the Map functions, the differences between the sub-Data Models will be found, then the Reduce functions will join the results. Notice that this is only a hypothesis which will need to be evaluated in the future.

Threats to Validity

As for internal validity, not all test cases were analyzed, i.e., only variability on device level (Logical Node) was evaluated and this may be a threat at the time of conclusion. The most variable part (device variability) has been evaluated considering the personal interviews with our industrial partners, but it would be interesting to examine other industrial domains to consider that it is also applicable to them.

As for external validity, evaluation has not been performed in a real environment, i.e. the computer used has fewer resources than a possible industrial PC. It would be, therefore, appropriate to perform this test in a real environment in order to obtain more realistic results.

6.3 Applying Solution into other domains

Due to the limited time of the thesis, it was not possible to evaluate the *modular monitoring system* and the *personalized visualization & evolution detection system* in other domains apart from AW. Therefore, once a concrete solution of *modular monitoring system*, i.e., *ICPSMonitoringProduct*, was evaluated in AW domain, an effort analysis was performed. In this manner, it was possible receive the opinion of engineers (i.e., developers, designers, testers) who used *ICPSMonitoringProduct* in terms of starting to monitor an ICPS that belongs to another domain. To do so, personal interviews were performed with engineers responsible of deploying *ICPSMonitoringProduct* into AW domain. Likewise, is possible to evaluate the cost of using *modular monitoring system* in other domains such as, CFTs and PLs.

Analyzing *modular monitoring system* it is noted that two main task need to be done in order to start monitoring an ICPS: (I) define the organizational structure of the

ICPS to be monitored considering the defined Data MetaModel; and (II) introduce the corresponding agents in order to create the communication between the monitoring system and the device. In terms of *personalized visualization & evolution detection system* it is noted that the use of it depends on the Data MetaModel, i.e., as concluded in the scalability test, the use of the system will depend on the quantity of devices within the ICPS and also if the Data MetaModel suits the necessities of the ICPS monitorization. That is why,it is interesting to analyze the applicability of the *modular monitoring system*. Thus, three questions were asked to engineers, i.e., developers, designers, and testers, who worked in the development of *ICPSMonitoringProduct*:

- **Structure Definition:** *"How long would it take to define a new logical or physical structure of an ICPS to start monitoring it in a new industrial Domain?"*
- **Agent Introduction:** *"How long would it take to introduce new agents to start monitoring an ICPS in other domains? "*
- **Other Comments:** *"Do you know anything else that needs to be taken into account when monitoring an ICPS or do you have any other comments?"*

In response to the first question and taking into account the response of the engineers involved in the deployment of *modular monitoring system*, non special effort needs to be done in order to define the structure of the ICPS to be monitored. As the MetaModel is already defined (i.e., IEC 61580 & IEC 62264), the user, independently of the industrial domain, is able to define the structure of the ICPS. No additional time is needed.

As for the second question, the engineers who participated in the deployment of *modular monitoring system* remarked that a considerable effort is needed. As already mentioned, in order to start monitoring an ICPS, in addition to defining the organizational structure of the ICPS, the corresponding agents must be introduced. This implies that as it is a new domain, no agent exists, as the agent repository is empty. Hence all the agents must be created from scratch.

For each type of existing device, an agent must be designed, developed and tested. That is why, all depends on the number of types of devices and even their own complexity. Therefore, for each case the time will depend on the agents to develop. With ULMA's experience, for each agent an average of 20 days is necessary since this must be designed, implemented and tested. However, once the agents exist, to start monitoring an ICPS does not take so much time (e.g. about 8 day in ULMA's case). Additionally, TRILATERAL (sof**T**ware p**R**oduct l**I**ne based mu**L**tidomain iot **A**r**T**ifact g**E**neration for indust**R**i**AL** cps), which can be used to create agents exists. Hence, in the future it would be interesting to analyze which is the return of investment of the tool [IIL+19b].

Finally, in response to the third question, the engineers notice that 50% to 90% of the agents are reused in AW. It is noted that some of the devices used in different domains are equal (e.g. temperature sensors), hence, a common repository between different domains (e.g., AW, CFT and PL) can become a good starting point in order to reduce time development and commissioning time. In this manner, agents can be shared between domains and reduce the agent development time when a new domain starts monitoring their ICPSs. Therefore, when a new ICPS needs to start monitoring, all the agents need to be introduced to the platform even if they are already created, hence, a manually work needs to be done. Thus, if the devices are autodiscovered and associated to the agent and the system is able to detect the evolution, an automatic configuration would be possible; making the configuration process less error-prone and faster.

PART V

FINAL REMARKS

"Focus on making yourself better, not on thinking that you are better."

BOHDI SANDERS

Chapter 7

Conclusion

This chapter concludes the dissertation. Specifically, it summarizes the contributions, where the hypothesis validation and the solution limitations are presented. Then, a set of lessons learned which were extracted from the dissertation are discussed. Finally, short and mid-term future work are exposed.

Contents

7.1 Summary of the Contributions **140**

7.1.1 Hypothesis Validation 141

7.1.2 Limitations 143

7.2 Lessons Learned **143**

7.3 Perspectives and Future Work **146**

7.3.1 Short-term Perspectives 146

7.3.2 Medium-term Perspectives 147

7.1 Summary of the Contributions

In this dissertation a *modular monitoring and adaptable & adaptive visualization solution for facing the evolution of Industrial Cyber-Physical Systems* is proposed.

Table 7.1: Identified Requirements VS Proposed solution

Requirements	**Modular Monitoring System**	**Personalized Visualization & Evolution Detection System**	
		Personalized Visualization Subsystem	**ICPS Evolution Detection Subsystem**
Dynamic Monitoring System	■		
Communication	■		
Structured Data	■		
Evolution Detection			■
Adaptable Visualization		■	
Adaptive Visualization		■	

First three different industrial domains are analyzed, Automated Warehouse (AW), Press Line (PL) and Catenary-Free Tram (CFT). After performing the domain analysis it is noted that commonalities exist between domains in terms of monitoring and visualizing their Industrial Cyber-Physical Systems (ICPSs). Those commonalities were converted into requirements and two main systems are presented to give response to those requirement. As shown in Table 7.1, the designed, developed and evaluated systems are:

- **Modular monitoring system:** it is possible to configure a monitoring system able to capture the variable data coming from the ICPS. In addition, it supports the ICPS evolution in order to store all data in a structured way based on different standards.

- **Personalized visualization & evolution detection system:** Gives the chance to the user to configure their dashboard with the corresponding Graphical Elements (GEs). In addition, it is capable of detecting and classifying ICPS evolution in an automatic way. Finally, the system is able to automatically adapt every dashboard and alert users about the evolution when necessary.

Both systems, which are connected with each other, have been individually tested. A prototype of each one was developed and considering our work some Questionnaire-Based Survery (QBS) and personal interviews were performed in order to obtain the return of investment considering technical managers opinion, and if the prototypes were of interest for different industrial domains. Additionally, considering the quantity of devices which can be connected, a scalability test was performed in order to prove how scalable our system is in terms of detecting and classifying ICPS evolution. However, as the prototypes were developed and validated with an AW domain use case, it was analyzed if our solution was suitable for different domains by conducting personal interviews with developers.

Considering the hypothesis exposed in Section 1.4 and the design, development and evaluation of the presented contributions, in Section 7.1.1 the hypothesis validation is presented followed by Section 7.1.2 where the solution limitations are exposed.

7.1.1 Hypothesis Validation

Three research hypotheses were stated in Section 1.4. This section analyses each of the contributions and argues whether the stated hypotheses can be validated.

Hypothesis 1

The first hypothesis is stated as follows: *"The use of models helps creating a unified solution across domains in order to monitor ICPSs"*. To test this hypothesis a system which is based on an abstraction of the identified requirements is proposed. That was possible due to the use of models. Additionally, a MetaModel, which was designed by the union of two standards, was defined in order to capture data from the devices. The defined system was then developed and validated with the industry. That is why, it can be said that this hypothesis has been contrasted.

Hypothesis 2

The second hypothesis is stated as follows: *"The use of tree models allows to detect and classify ICPS evolution"*. To test this hypothesis a system able to compare at runtime subsequent tree-models (Data Model) in order to (I) detect evolution; and (II) classify the evolution to alert the end user besides adapting the User Interface (UI); was designed and developed. In addition, an scalability test was performed in order to test the algorithm in different ICPSs sizes. Thanks to the scalability test it can be concluded that the proposed system is more appropriated to small Data Models, since the greater the model to be compared, the greater the time needed, this being a disadvantage to generate alerts in large Data Models. For this reason, it is proposed to improve the system by means of parallel computing theory. In spite of the difficulty found in the system, i.e. time plays against it when the models are large, it has been possible to validate the hypothesis because, thanks to the tree models structure, it is possible to efficiently detect and classify ICPS evolution.

Hypothesis 3

The third hypothesis is stated as follows: *"Adaptive and Adaptable systems help on decision making when the ICPS evolves"*. To test this hypothesis a system was designed, developed and evaluated, i.e., a *personalized visualization & evolution detection system*. This system is able to: (I) automatically adapt the visualization based on the ICPS evolution (adaptive system); and (II) give the chance to the end user to create and update UI considering the data captured from the devices (adaptable system). After designing the system, this one was developed (prototype) and evaluated where a return of investment and a QBS was performed. It is noted that benefits of the proposed systems will grow along time and considering the results of the QBS the created prototype is interesting for the users. Additionally, it is important to mention that it also covers most of the ideas depicted in the Big Data Value Association (BDVA) [BCJP16]. Therefore, because of the evaluation and the requirements set by the BDVA, it is considered that adapting the UI to the user and evolving this one when the ICPS evolves is a step forward for the industry. That is why, it can be said that this hypothesis has been contrasted.

7.1.2 Limitations

This section discusses the limitations that the proposed systems have:

- **Data capture frequency**: When the agents are deployed, the devices start sending data and the monitoring system is the responsible for capturing all the data and performing the corresponding Data Model. The biggest limitation of the proposed system is the frequency since, if the devices start sending data in different frequency, the monitoring system will create different Data Models which can provoke confusion, i.e., the system can detect an evolution due to the frequency difference.
- **Data MetaModel**: A Data MetaModel is designed with the combination of two different standards. Even if an evaluation of this Data MetaModel is presented, the following points must be taken into account in order to use this Data MetaModel:
  - **Physical / logical structure**: Even if in an ICPS a node can communicate with other nodes, in our Data MetaModel the relation between nodes is not reflected. Each node is independent of the rest.
  - **Atomic values**: Although the Data MetaModel supports complex data structures, this dissertation in not focused on the analysis of them. That is why, the Data MetaModel is designed for atomic data values, i.e., simple data (e.g., Boolean, Integer, String).
- **Evolution detection system**: An *evolution detection system* is presented as part of the *personalized visualization & evolution detection system*. With this, it is possible to detect different kind of ICPS evolution, but the biggest limitation of it is that it is not currently able to give response the big ICPS sizes. Therefore, as future work, it seems promising to improve the proposed system by introducing parallel computing theory.

7.2 Lessons Learned

This section presents lessons learned during the design, development and evaluation of the concrete *monitoring software system* and the *personalized visualization &*

evolution detection system.

Models Enable To Understand Common Challenges and Devise a Generic Solution

In industry, the focus is on implementing a solution right away in a group due to shorter deadlines to deliver products and services. Such a solution may not be effective in a long run, especially if the solution is targeting a complex problem across different systems being designed, developed, and implemented in different groups. This requires understanding the problem at a higher level of abstraction (i.e., using models) from different systems and developing a unified solution (i.e., based on model-based engineering) that is applicable across the systems. This was the case of different industrial domains i.e., AW, PL and CFT. All of these systems require a *modular monitoring system* besides a *personalized visualization & evolution detection system* in order to supervise their ICPSs. Thus, commonalities across multiple systems to devise a generic solution were identified. Thus, models played a crucial role in understanding the challenges and managing the overall complexity of the problem at hand via abstraction. Furthermore, by abstracting the challenges at a higher level of abstraction, a solution for *the management of evolving ICPSs in terms of monitoring and visualization* was devised, i.e., models facilitated to provide an automated solution to the problem.

Models Provide a Basis for an Efficient Solution for Multiple Clients and Different Systems

Taking ULMA company as an example, due to the use of models, the same product designed and developed for ULMA's clients can be used by ULMA. The use of models enabled adapting the monitoring system besides the visualization for different scenarios in a personalized way and even facilitated comparisons between their clients for preventing incidents. Thus, using models and combining them with international standards, makes the designed and developed systems be used by: (I) different clients from the same domain (e.g., Intermarché , Cárnicas Tello); and (II) clients who are dedicated to the design, development, and maintenance of industrial domains (e.g., ULMA), even though in a different industrial domain such as the PL domain or CFT e.g., Fagor Arrasate with its client ArcelorMittal.

Models Enable Designing a Solution Independent from Technologies

Although the development of the eventual solution required an understanding of technologies for implementation, model-based engineering facilitated the design of the system independent of the technologies, i.e., platform independent model (PIM). Our solution was intended for different industrial domains which have different ICPSs, hence, the use of models helped us to abstract a solution independent of technical details, i.e., how to capture data (*modular monitoring system*), and how to visualize the captured data (*personalized visualization & evolution detection system*).

More specifically, *modular monitoring system* does not care about which is the Internet of Things (IoT) communication protocol used by the agent. Thanks to the use of agent and the designed Data MetaModel, the defined monitoring system (*ICPSMonitoringProduct*) does not care about how the communication is performed, giving to each customer the chance of deciding the technology that best suits. In addition, the *personalized visualization & evolution detection system* design was independent of various technologies such as Infogram or libraries such as D3 or HighCharts. Doing so the effort to deploy *personalized visualization & evolution detection system* in a new domain is reduced.

Concerning technologies, new perspectives were learned. It was noted that using external APIs (i.e., Infogram) is risky due to its tight dependency on third-party components. Such dependencies are not desirable in the industry. Relying on external APIs makes the quality of service provided by a system dependent on the third-party components. If Infogram in our context stops working, the *personalized visualization & evolution detection system* will not generate graphical elements or visualize existing ones. Therefore, it is best to use libraries that can be embedded in the *personalized visualization & evolution detection system*, such as D3 or HighChart, i.e., internal services. Another option would be to develop in-house libraries for this purpose.

Agents to Provide ICPS Evolution and Data Structures to Manage it

In terms of *modular monitoring system*, the potential of agents was learned. When an ICPS needs to be monitored, the evolution that this one can have is not usually considered, hence, it makes the adaptation of the monitoring system and the visualization error-prone. That is why, changing the way of thinking and making each element independent (i.e., agent) give as the chance of managing evolution at runtime.

Additionally, thinking about a common data structure which can envisage evolution gives us the chance for analyzing evolution and almost automatically adapting the whole vertical, i.e., from the data capture to its visualization.

Using Standards for Give a Common Solution

In this dissertation, three different standards are used in order to adapt the design and develop a prototype valid for different domains. It is noted that the use of standards have the following advantages: (I) interoperability; and (II) intuitive device and data modeling and naming, using hierarchical structure, instead of plain format. Thus, using standards in industry in order to give a common solution is a good point is important.

7.3 Perspectives and Future Work

In this section the short and medium term objectives to complement the realized work are summarized, since even if the results obtained from this research can be considered useful, there are still several areas that can be further explored.

7.3.1 Short-term Perspectives

Several areas are still open for improvement. Here below some of the points that can improve the proposed systems in the short term are presented.

- **From prototypes to Industry**. Even if the proposed research solution is industry oriented, it would be interesting to transfer both systems to the real world. In this manner, a more precise evaluation can be done and the efficiency of using models would be tested in a real environment.
- **Improve evolution detection and classification subsystem.** Considering the execution time increases sup-linearly when the ICPS data model size increases (See Chapter 6), it would be interesting to improve the subsystem. The hypothesis says that, if the size of the input files is decreased and parallel computing theory is used, it would be possible to decrease the total execution time, enabling new time scenarios.

- **Add artificial intelligence techniques to propose visualizations**. Considering every user can create each own visualizations, analyzing how each one usually visualizes data and proposing visualizations at runtime would be interesting. In this manner, the system itself can help the user on visualizing data and hence, on decision making. To do so, it would be interesting to explore artificial intelligence techniques such as Reinforcement Learning.

7.3.2 Medium-term Perspectives

In order to further improve the proposed solution, additional perspectives in the long-term are also presented.

- **Automatic propagation when agents are updated**. Even if an ICPS is composed by different devices, the same agent can be used for more than one device. Thus, when an agent is updated, propagating this change into all the agents (which are equal) in automatic way can improve our system, since currently, it is the user the one who needs to change every existent agent. Along with this, and considering the created tool (TRILATERAL) in collaboration with another PhD student, it would be interesting to improve the tool in order to be able to create drivers in automatic way. In this manner, the tool and our system can be joined and the user does not need to introduce every created agent to the system.

- **Device self-discovery**. Note that if a new device is introduced in an ICPS, the task of monitoring the new device is supposed to be allocated to an agent that should be associated with a platform. Both (agent, platform) can be either existing already or newly created. However, in our system, the operator needs to manage those changes manually in the ICPSMonitoringProduct to monitor the new device. Manually managing the changes has shown some drawbacks. First, when changes happen frequently, it is a burden for the operator. Secondly, the manual process is error-prone. For instance, if the operator does not manually associate an agent with a platform, data from the device cannot be received. Thus, it would be interesting to improve the modular monitoring by detecting automatically devices within ICPS and associating them automatically to an agent without human intervention. To do so, using SWIM IoT communication protocol [DGM02] looks promising since Orive et al. [OABM18] use this protocol to know the existence of a node when it falls or rises in a system. Thanks to the communication between them, they can detect if the nodes are alive or not. That is why if there is communication between the devices using this prototype, it is possible to use it for discovering them. This new idea should be further analyzed because to detect a device through this protocol there must be minimal interaction with another device. In addition, this protocol has been evaluated on a large scale by Snyder et al. [SCJ+14], which makes it possible to use it with many devices.

Bibliographic References

[61211] RFC 6121. Extensible messaging and presence protocol (xmpp): Instant messaging and presence, 2011.

[ABBB15] Roberto Acerbis, Aldo Bongio, Marco Brambilla, and Stefano Butti. Model-driven development based on omg's IFML with webratio web and mobile platform. In *Engineering the Web in the Big Data Era - 15th International Conference, ICWE 2015, Rotterdam, The Netherlands, June 23-26, 2015, Proceedings*, pages 605–608, 2015.

[ABCC05] Francesco Amato, Francesco Basile, Ciro Carbone, and Pasquale Chiacchio. An approach to control automated warehouse systems. *Control Engineering Practice*, 13(10):1223 – 1241, 2005.

[Abe13] Andrew Abela. *Advanced Presentations by Design: Creating Communication that Drives Action*. Wiley, 2013.

[ABY14a] Pierre A. Akiki, Arosha K. Bandara, and Yijun Yu. Adaptive model-driven user interface development systems. *ACM Comput. Surv.*, 47(1):9:1–9:33, 2014.

[ABY14b] Pierre A. Akiki, Arosha K. Bandara, and Yijun Yu. Integrating adaptive user interface capabilities in enterprise applications. In *36th International Conference on Software Engineering, ICSE '14, Hyderabad, India - May 31 - June 07, 2014*, pages 712–723, 2014.

[ANS05] Ansi/isa-95.00.03-2005, enterprise-control system integration, part 3: Models of manufacturing operations management, 2005.

[ASB+09] Vander Alves, Daniel Schneider, Martin Becker, Nelly Bencomo, and Paul Grace. Comparitive study of variability management in software product lines and runtime adaptable systems. In *Third International*

Workshop on Variability Modelling of Software-Intensive Systems, Seville, Spain, January 28-30, 2009. Proceedings, pages 9–17, 2009.

[ASFV18] Thomas Aicher, Markus Spindler, Johannes Fottner, and Birgit Vogel-Heuser. Analyzing the industrial scalability of backwards compatible intralogistics systems. *Production Engineering*, 12(3-4):297–307, 2018.

[AWSE16] Aitor Arrieta, Shuai Wang, Goiuria Sagardui, and Leire Etxeberria. Search-based test case selection of cyber-physical system product lines for simulation-based validation. In *Proceedings of the 20th International Systems and Software Product Line Conference*, pages 297–306. ACM, 2016.

[BBA+17] Ada Bagozi, Devis Bianchini, Valeria De Antonellis, Alessandro Marini, and Davide Ragazzi. Big data summarisation and relevance evaluation for anomaly detection in cyber physical systems. In *On the Move to Meaningful Internet Systems. OTM 2017 Conferences - Confederated International Conferences: CoopIS, C&TC, and ODBASE 2017, Rhodes, Greece, October 23-27, 2017, Proceedings, Part I*, pages 429–447, 2017.

[BBCW18] Hugo Brunelière, Erik Burger, Jordi Cabot, and Manuel Wimmer. A feature-based survey of model view approaches. In *Proceedings of the 21th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, MODELS 2018, Copenhagen, Denmark, October 14-19, 2018*, page 211, 2018.

[BCJP16] Tilman Becker, Edward Curry, Anja Jentzsch, and Walter Palmetshofer. New horizons for a data-driven economy: Roadmaps and action plans for technology, businesses, policy, and society., 2016.

[BLA10] Marco Blumendorf, Grzegorz Lehmann, and Sahin Albayrak. Bridging models and systems at runtime to build adaptive user interfaces. In *Proceedings of the 2nd ACM SIGCHI Symposium on Engineering Interactive Computing System, EICS 2010, Berlin, Germany, June 19-23, 2010*, pages 9–18, 2010.

[BMC+17] Safa Bougouffa, Kilian Meßzmer, Suhyun Cha, Emanuel Trunzer, and Birgit Vogel-Heuser. Industry 4.0 interface for dynamic reconfiguration of an open lab size automated production system to allow remote community experiments. In *2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 2058–2062, Dec 2017.

[Bo10] Yang Bo. Querying json streams, 2010.

[BQ15] Luciano Baresi and Clément Quinton. Dynamically evolving the structural variability of dynamic software product lines. In *10th IEEE/ACM International Symposium on Software Engineering for Adaptive and Self-Managing Systems, SEAMS 2015, Florence, Italy, May 18-19, 2015*, pages 57–63, 2015.

[CAM02] Gregory Cobena, Serge Abiteboul, and Amélie Marian. Detecting changes in XML documents. In *Proceedings of the 18th International Conference on Data Engineering, San Jose, CA, USA, February 26 - March 1, 2002*, pages 41–52, 2002.

[Che14] Xi Chen. Constrained application protocol for internet of things. *URL: https://www. cse. wustl. edu/~jain/cse574-14/ftp/coap*, 2014.

[CLW+16] Huina Chao, Huawei Li, Tiancheng Wang, Xiaowei Li, and Bo Liu. An accurate algorithm for computing mutation coverage in model checking. In *2016 IEEE International Test Conference, ITC 2016, Fort Worth, TX, USA, November 15-17, 2016*, pages 1–10, 2016.

[CZH17] Zhehan Chen, Xiaohua Zhang, and Ketai He. Research on the technical architecture for building CPS and its application on a mobile phone factory. In *5th International Conference on Enterprise Systems, ES 2017, Beijing, China, September 22-24, 2017*, pages 76–84, 2017.

[DEJE16] Antonio Delgado, Antonio Jose Estepa, José Antonio Troyano Jiménez, and Rafael Estepa. Reusing UI elements with model-based user interface development. *Int. J. Hum.-Comput. Stud.*, 86:48–62, 2016.

[DGM02] Abhinandan Das, Indranil Gupta, and Ashish Motivala. SWIM: scalable weakly-consistent infection-style process group membership protocol. In *2002 International Conference on Dependable Systems and Networks (DSN 2002), 23-26 June 2002, Bethesda, MD, USA, Proceedings*, pages 303–312, 2002.

[dR17] Oscar Peña del Rio. *A Data-Driven Visual Approach to Explore Linked Open Data Environments*. PhD thesis, Universidad de Deusto, Facultad de Ingeniería, 2017.

[FBT18] Yasmin Fathy, Payam Barnaghi, and Rahim Tafazolli. Large-scale indexing, discovery, and ranking for the internet of things (iot). *ACM Comput. Surv.*, 51(2):29:1–29:53, March 2018.

[Fie00] Roy Thomas Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, 2000. AAI9980887.

[FKF+16] Hans Fleischmann, Johannes Kohl, Jörg Franke, Andreas Reidt, Markus Duchon, and Helmut Krcmar. Improving maintenance processes with distributed monitoring systems. In *14th IEEE International Conference on Industrial Informatics, INDIN 2016, Poitiers, France, July 19-21, 2016*, pages 377–382, 2016.

[GBH+16] Ilias Gerostathopoulos, Tomás Bures, Petr Hnetynka, Jaroslav Keznikl, Michal Kit, Frantisek Plasil, and Noël Plouzeau. Self-adaptation in software-intensive cyber-physical systems: From system goals to architecture configurations. *Journal of Systems and Software*, 122:378–397, 2016.

[GFSV14] Alfonso García Frey, Jean-Sébastien Sottet, and Alain Vagner. AME: an adaptive modelling environment as a collaborative modelling tool. In *ACM SIGCHI Symposium on Engineering Interactive Computing Systems, EICS'14, Rome, Italy, June 17-20, 2014*, pages 189–192, 2014.

[GPM14] Marisol García-Valls, Diego Perez-Palacin, and Raffaela Mirandola. Time-sensitive adaptation in CPS through run-time configuration generation and verification. In *IEEE 38th Annual Computer Software and Applications Conference, COMPSAC 2014, Vasteras, Sweden, July 21-25, 2014*, pages 332–337, 2014.

[HB15] Martin Hankel and Rexroth Bosch. The reference architectural model industrie 4.0 (rami 4.0). *ZVEI*, 2015.

[HBS+02] Mark Hapner, Rich Burridge, Rahul Sharma, Joseph Fialli, and Kate Stout. Java message service. *Sun Microsystems Inc., Santa Clara, CA*, 9, 2002.

[HR16] Marco E. Perez Hernandez and Stephan Reiff-Marganiec. Towards a software framework for the autonomous internet of things. In *4th IEEE International Conference on Future Internet of Things and Cloud, FiCloud 2016, Vienna, Austria, August 22-24, 2016*, pages 220–227, 2016.

[Hu16] GQ Hu. Real-time monitoring system of hazardous chemicals based on 6lowpan and wlan. *Chemical Engineering Transactions*, 51:139–144, 2016.

[ICMU18a] Markel Iglesias-Urkia, Diego Casado-Mansilla, Simon Mayer, and Aitor Urbieta. Enhanced publish/subscribe in coap: describing advanced subscription mechanisms for the observe extension. In *Proceedings of the 8th International Conference on the Internet of Things, IOT 2018*, pages 14:1–14:8, 2018.

[ICMU18b] Markel Iglesias-Urkia, Diego Casado-Mansilla, Simon Mayer, and Aitor Urbieta. Validation of a coap to IEC 61850 mapping and benchmarking vs HTTP-REST and WS-SOAP. In *23rd IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2018*, pages 1015–1022, 2018.

[IEC13a] IEC TR 61850-1:2013 | IEC Webstore | LVDC, 2013.

[IEC13b] IEC 62264-1:2013 - Enterprise-control system integration – Part 1: Models and terminology, 2013.

[IIL$^+$19a] Aitziber Iglesias., Markel Iglesias-Urkia., Beatriz López-Davalillo., Santiago Charramendieta., and Aitor Urbieta. Trilateral: Software product line based multidomain iot artifact generation for industrial cps. In *Proceedings of the 7th International Conference on Model-Driven Engineering and Software Development - Volume 1: MODELSWARD,*, pages 64–73. INSTICC, SciTePress, 2019.

[IIL$^+$19b] Markel Iglesias-Urkia., Aitziber Iglesias., Beatriz López-Davalillo., Santiago Charramendieta., Diego Casado-Mansilla., Goiuria Sagardui., and Aitor Urbieta. Trilateral: a model-based approach for industrial cps - monitoring and control. In *Communications in Computer and Information Science (To be published)*. Springer International Publishing, 2019.

[ILA+17] Aitziber Iglesias, Hong Lu, Cristóbal Arellano, Tao Yue, Shaukat Ali, and Goiuria Sagardui. Product line engineering of monitoring functionality in industrial cyber-physical systems: A domain analysis. In *Proceedings of the 21st International Systems and Software Product Line Conference, SPLC 2017, Volume A, Sevilla, Spain, September 25-29, 2017*, pages 195–204, 2017.

[ISA19] Aitziber Iglesias, Goiuria Sagardui, and Cristobal Arellano. Industrial cyber-physical system evolution detection and alert generation. *Applied Sciences*, 9(8), 2019.

[IUCMMU18] Markel Iglesias-Urkia, Diego Casado-Mansilla, Simon Mayer, and Aitor Urbieta. Integrating electrical substations within the iot using iec 61850, coap and cbor. *Submitted to IEEE Internet of Things Journal*, 00 2018.

[IUUPCM17] Markel Iglesias-Urkia, Aitor Urbieta, Jorge Parra, and Diego Casado-Mansilla. IEC 61850 meets coap: towards the integration of smart grids and iot standards. In *Proceedings of the Seventh International Conference on the Internet of Things, IOT 2017*, pages 3:1–3:9, 2017.

[IYA+18] Aitziber Iglesias, Tao Yue, Cristóbal Arellano, Shaukat Ali, and Goiuria Sagardui. Model-based personalized visualization system for monitoring evolving industrial cyber-physical system. In *25th Asia-Pacific Software Engineering Conference, APSEC 2018, Nara, Japan, December 4-7, 2018*, pages 532–541, 2018.

[JHZ+15] Benben Jiang, Dexian Huang, Xiaoxiang Zhu, Fan Yang, and Richard D Braatz. Canonical variate analysis-based contributions for fault identification. *Journal of Process Control*, 26:17–25, 2015.

[JN11] S. Julie and I. Noah. *Designing Data Visualizations*. O'Reilly Media, Inc., 2011.

[KBB+09] Barbara A. Kitchenham, Pearl Brereton, David Budgen, Mark Turner, John Bailey, and Stephen G. Linkman. Systematic literature reviews in software engineering - A systematic literature review. *Information & Software Technology*, 51(1):7–15, 2009.

[Kee07] Staffs Keele. Guidelines for performing systematic literature reviews in software engineering. Technical report, Technical report, Ver. 2.3 EBSE Technical Report. EBSE, 2007.

[KHHW13] Hernning Kagermann, Johannes Helbig, Ariane Hellinger, and Wolfgang Wahlster. *Recommendations for Implementing the Strategic Initiative INDUSTRIE 4.0: Securing the Future of German Manufacturing Industry ; Final Report of the Industrie 4.0 Working Group*. Forschungsunion, 2013.

[KL10] Richard Kennard and John Leaney. Towards a general purpose architecture for UI generation. *Journal of Systems and Software*, 83(10):1896–1906, 2010.

[KVA13] Hermann Kaindl, Mathieu Vallee, and Edin Arnautovic. Self-representation for self-configuration and monitoring in agent-based flexible automation systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 43(1):164–175, Jan 2013.

[LBA10] Grzegorz Lehmann, Marco Blumendorf, and Sahin Albayrak. Development of context-adaptive applications on the basis of runtime user interface models. In *Proceedings of the 2nd ACM SIGCHI Symposium on Engineering Interactive Computing System, EICS 2010, Berlin, Germany, June 19-23, 2010*, pages 309–314, 2010.

[LBFA08] Grzegorz Lehmann, Marco Blumendorf, Sebastian Feuerstack, and Sahin Albayrak. Utilizing dynamic executable models for user interface development. In *Interactive Systems. Design, Specification, and Verification, 15th International Workshop, DSV-IS 2008, Kingston, Canada, July 16-18, 2008, Revised Papers*, pages 306–309, 2008.

[LBK15] Jay Lee, Behrad Bagheri, and Hung-An Kao. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing Letters*, 3:18 – 23, 2015.

[LCK16] Paulo Leitão, Armando Walter Colombo, and Stamatis Karnouskos. Industrial automation based on cyber-physical systems technologies: Prototype implementations and challenges. *Computers in Industry*, 81:11–25, 2016.

[Lee08] Edward A. Lee. Cyber physical systems: Design challenges. In *11th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC 2008), 5-7 May 2008, Orlando, Florida, USA*, pages 363–369, 2008.

[LH12] X. Lu and M. Huang. System-decomposition-based multilevel control for hydraulic press machine. *IEEE Trans. Industrial Electronics*, 59(4):1980–1987, 2012.

[LHZ+16] Guangzhu Liu, LiQun Hu, GuoQiang Zhong, Kai Li, RuiJie Zhou, and Neng Pu. Labview-based radiation monitoring system of east. *Journal of Fusion Energy*, 35(2):470–481, 2016.

[LKY14a] Jay Lee, Hung-An Kao, and Shanhu Yang. Service innovation and smart analytics for industry 4.0 and big data environment. *Procedia CIRP*, 16:3 – 8, 2014.

[LKY14b] Jay Lee, Hung-An Kao, and Shanhu Yang. Service innovation and smart analytics for industry 4.0 and big data environment. *Procedia CIRP*, 16:3 – 8, 2014.

[Loc10] Dave Locke. Mq telemetry transport (mqtt) v3. 1 protocol specification. *IBM developerWorks Technical Library*, page 15, 2010.

[LR02] Victor B Lortz and Jonathan G Ritchie. Device handling system and method, July 9 2002. US Patent 6,418,486.

[LX03] Haifeng Lu and Fangqin Xu. Container production management system based on the cps. In *2014 International Conference on Future Computer and Communication Engineering (ICFCCE 2014)*. Atlantis Press, 2014/03.

[MCS14] Miroslav Macik, Tomás Cerný, and Pavel Slavík. Context-sensitive, cross-platform user interface generation. *J. Multimodal User Interfaces*, 8(2):217–229, 2014.

[MDCTS14] Aitor Murguzur, Xabier De Carlos, Salvador Trujillo, and Goiuria Sagardui. *Context-Aware Staged Configuration of Process Variants@Runtime*, pages 241–255. Springer International Publishing, 2014.

[MGR02] Sergey Melnik, Hector Garcia-Molina, and Erhard Rahm. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In *Proceedings of the 18th International Conference on Data Engineering, San Jose, CA, USA, February 26 - March 1, 2002*, pages 117–128, 2002.

[MLD09] Wolfgang Mahnke, Stefan-Helmut Leitner, and Matthias Damm. *OPC unified architecture*. Springer Science & Business Media, 2009.

[MMKV15] Nesrine Mezhoudi, Jorge Luis Perez Medina, Iyad Khaddam, and Jean Vanderdonckt. Context-awareness meta-model for user interface runtime adaptation. *International Journal of Software Engineering*, 2015.

[MNSY18] Jacopo Mauro, Michael Nieke, Christoph Seidl, and Ingrid Chieh Yu. Context-aware reconfiguration in evolving software product lines. *Sci. Comput. Program.*, 163:139–159, 2018.

[MPFM08] Shahid Mujtaba, Kai Petersen, Robert Feldt, and Michael Mattsson. Software product line variability: A systematic mapping study. *School of Engineering, Blekinge Inst. of Technology*, 2008.

[MRWH15] Anitha Murugesan, Sanjai Rayadurgam, Michael W. Whalen, and Mats Per Erik Heimdahl. Design considerations for modeling modes in cyber-physical systems. *IEEE Design & Test*, 32(5):66–73, 2015.

[NBK+15] Oliver Niggemann, Gautam Biswas, John S. Kinnebrew, Hamed Khorasgani, Sören Volgmann, and Andreas Bunte. Data-driven monitoring of cyber-physical systems leveraging on big data and the internet-of-things for diagnosis and control. In *Proceedings of the 26th International Workshop on Principles of Diagnosis (DX-2015) co-located with 9th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes (Safeprocess 2015), Paris, France, August 31 - September 3, 2015.*, pages 185–192, 2015.

[OABM18] Adrian Orive, Aitor Agirre, Josu Bilbao, and Marga Marcos. Passive network state monitoring for dynamic resource management in industry 4.0 fog architectures. In *14th IEEE International Conference on Automation Science and Engineering, CASE 2018, Munich, Germany, August 20-24, 2018*, pages 1414–1419, 2018.

[OMG15] OMG. Data distribution service (dds), April 2015.

[PBM07] Hervé Panetto, Salah Baïna, and Gérard Morel. Mapping the iec 62264 models onto the zachman framework for analysing products information traceability: a case study. *Journal of Intelligent Manufacturing*, 18(6):679–698, 2007.

[PDB13] Fabio Pasqualetti, Florian Dörfler, and Francesco Bullo. Attack detection and identification in cyber-physical systems. *IEEE Trans. Automat. Contr.*, 58(11):2715–2729, 2013.

[PHA+10] Anders Bro Pedersen, Einar Bragi Hauksson, Peter Bach Andersen, Bjarne Poulsen, Chresten Traeholt, and Dieter Gantenbein. Facilitating a generic communication interface to distributed energy resources: Mapping iec 61850 to restful services. In *2010 First IEEE International Conference on Smart Grid Communications*, pages 61–66, Oct 2010.

[PKKP10] Mi Jeong Park, Dong Kwan Kim, Won-Tae Kim, and Seung-Min Park. Dynamic software updates in cyber-physical systems. In *2010 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 425–426, Nov 2010.

[PWA13] Cesare Pautasso, Erik Wilde, and Rosa Alarcon. *REST: advanced research topics and practical applications*. Springer, 2013.

[QRV+15] Clément Quinton, Rick Rabiser, Michael Vierhauser, Paul Grünbacher, and Luciano Baresi. Evolution in dynamic software product lines: challenges and perspectives. In *Proceedings of the 19th International Conference on Software Product Line, SPLC 2015, Nashville, TN, USA, July 20-24, 2015*, pages 126–130, 2015.

[Ran10] David Raneburger. Interactive model driven graphical user interface generation. In *Proceedings of the 2nd ACM SIGCHI Symposium on Engineering Interactive Computing System, EICS 2010, Berlin, Germany, June 19-23, 2010*, pages 321–324, 2010.

[RJHA18] Tamás Ruppert, Szilárd Jaskó, Tibor Holczinger, and János Abonyi. Enabling technologies for operator 4.0: A survey. *Applied Sciences*, 8(9):1650, 2018.

[RLSS10] Ragunathan Rajkumar, Insup Lee, Lui Sha, and John A. Stankovic. Cyber-physical systems: the next computing revolution. In *Proceedings of the 47th Design Automation Conference, DAC 2010, Anaheim, California, USA, July 13-18, 2010*, pages 731–736, 2010.

[RSY+17] Thomas Brox Røst, Christoph Seidl, Ingrid Chieh Yu, Ferruccio Damiani, Einar Broch Johnsen, and Cristina Chesta. Hyvar - scalable hybrid variability for distributed evolving software systems. In *Advances in Service-Oriented and Cloud Computing - Workshops of ESOCC 2017, Oslo, Norway, September 27-29, 2017, Revised Selected Papers*, pages 159–163, 2017.

[SA05] Peter Saint-Andre. Streaming xml with jabber/xmpp. *IEEE Internet Computing*, 9(5):82–89, September 2005.

[SC12] Matthew Stephan and James R. Cordy. A survey of methods and applications of model comparison technical report 2011-582 rev . 2, 2012.

[SC13] Matthew Stephan and James R. Cordy. A survey of model comparison approaches and applications. In *MODELSWARD 2013 - Proceedings of the 1st International Conference on Model-Driven Engineering and Software Development, Barcelona, Spain, 19 - 21 February, 2013*, pages 265–277, 2013.

[SCC+17] Kunal Suri, Arnaud Cuccuru, Juan Cadavid, Sebastien Gerard, Walid Gaaloul, and Samir Tata. Model-based development of modular complex systems for accomplishing system integration for industry 4.0. In *Proceedings of the 5th International Conference on Model-Driven Engineering and Software Development - MODELSWARD,*, pages 487–495. INSTICC, SciTePress, 2017.

[SCJ+14] Shane Snyder, Philip H. Carns, Jonathan Jenkins, Kevin Harms, Robert B. Ross, Misbah Mubarak, and Christopher D. Carothers. A case for epidemic fault detection and group membership in HPC storage systems. In *High Performance Computing Systems. Performance Modeling, Benchmarking, and Simulation - 5th International Workshop, PMBS 2014, New Orleans, LA, USA, November 16, 2014. Revised Selected Papers*, pages 237–248, 2014.

[SCPR06] Wolfgang Stürzlinger, Olivier Chapuis, Dusty Phillips, and Nicolas Roussel. User interface façades: towards fully adaptable user interfaces. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology, Montreux, Switzerland, October 15-18, 2006*, pages 309–318, 2006.

[SCZ+16] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646, 2016.

[SDSS16] Ljiljana Stojanovic, Marko Dinic, Nenad Stojanovic, and Aleksandar Stojadinovic. Big-data-driven anomaly detection in industry (4.0): an approach and a case study. In *2016 IEEE International Conference on Big Data, BigData 2016, Washington DC, USA, December 5-8, 2016*, pages 1647–1652, 2016.

[SHSH09] SK Subramaniam, SH Husin, RSS Singh, and AH Hamidon. Production monitoring system for monitoring the industrial shop floor performance. *International journal of systems applications, engineering & development*, 3(1):28–35, 2009.

[SPHV10] Arne Schramm, André Preußner, Matthias Heinrich, and Lars Vogel. Rapid UI development for enterprise applications: Combining manual and model-driven techniques. In *Model Driven Engineering Languages and Systems - 13th International Conference, MODELS 2010, Oslo, Norway, October 3-8, 2010, Proceedings, Part I*, pages 271–285, 2010.

[SPO+17] Andreas Daniel Sinnhofer, Peter Pühringer, Felix Jonathan Oppermann, Klaus Potzmader, Clemens Orthacker, Christian Steger, and Christian Kreiner. Combining business process variability and software variability using traceable links. In *Business Modeling and Software Design - 7th International Symposium, BMSD 2017, Revised Selected Papers*, pages 67–86, 2017.

[SS08] Marco Sebastiani and Panon Supiratana. Tracing the differences on an evolving software model, 2008.

[Sta12] OASIS Standard. Oasis advanced message queuing protocol (amqp) version 1.0. *International Journal of Aerospace Engineering Hindawi www. hindawi. com*, 2018, 2012.

[TAS12] Clarence JM Tauro, Shreeharsha Aravindh, and AB Shreeharsha. Comparative study of the new generation, agile, scalable, high performance nosql databases. *International Journal of Computer Applications*, 48(20):1–4, 2012.

[TC-03] IEC TC-57. Communication networks and systems in substations - part 7-1: Basic communication structure for substation and feeder equipment - principles and models, 2003.

[TFK+18] Akihiro Tsuchiya, Francisco Fraile, Ichiro Koshijima, Angel Órtiz, and Raúl Poler. Software defined networking firewall for industry 4.0 manufacturing systems. *Journal of Industrial Engineering and Management (JIEM)*, 11(2):318–333, 2018.

[TFPE12] Santiago J. Barro Torres, Tiago M. Fernández-Caramés, Héctor J. Pérez-Iglesias, and Carlos J. Escudero. Real-time personal protective equipment monitoring system. *Computer Communications*, 36(1):42–50, 2012.

[TLWD18] Hao Tang, Di Li, Shiyong Wang, and Zhijie Dong. CASOA: an architecture for agent-based manufacturing system in the context of industry 4.0. *IEEE Access*, 6:12746–12754, 2018.

[TVY90] Hideaki Takeda, Paul Veerkamp, and Hiroyuki Yoshikawa. Modeling design process. *AI magazine*, 11(4):37–37, 1990.

[TZXZ14] Fei Tao, Ying Zuo, Li Da Xu, and Lin Zhang. Iot-based intelligent perception and access of manufacturing resource toward cloud manufacturing. *IEEE Trans. Industrial Informatics*, 10(2):1547–1557, 2014.

[VK04] Vijay K. Vaishnavi and William L. Kuechler. Design Science Research in Information Systems. *Ais*, pages 1–45, 2004.

[WC00] Michael Wooldridge and Paolo Ciancarini. Agent-oriented software engineering: The state of the art. In *Agent-Oriented Software Engineering, First International Workshop, AOSE 2000, Limerick, Ireland, June 10, 2000, Revised Papers*, pages 1–28, 2000.

[WMO+16] Stephan Weyer, Torben Meyer, Moritz Ohmer, Dominic Gorecky, and Detlef Zühlke. Future modeling and simulation of cps-based factories: an example from the automotive industry. *IFAC-PapersOnLine*, 49(31):97 – 102, 2016. 12th IFAC Workshop on Intelligent Manufacturing Systems IMS 2016.

[Woo97] Michael Wooldridge. Agent-based software engineering. *Software Engineering - IEE Proceedings*, 144(1):26–37, 1997.

[WSBR12] Gabriel A. Weaver, Sean W. Smith, Rakesh B. Bobba, and Edmond J. Rogers. Re-engineering grep and diff for nerc cip. In *2012 IEEE Power and Energy Conference at Illinois*, pages 1–8, Feb 2012.

[XS05] Zhenchang Xing and Eleni Stroulia. Umldiff: an algorithm for object-oriented design differencing. In *20th IEEE/ACM International Conference on Automated Software Engineering (ASE 2005), November 7-11, 2005, Long Beach, CA, USA*, pages 54–65, 2005.

[ZJP+18] Xi Zheng, Christine Julien, Rodion M. Podorozhny, Franck Cassez, and Thierry Rakotoarivelo. Efficient and scalable runtime monitoring for cyber-physical system. *IEEE Systems Journal*, 12(2):1667–1678, 2018.

[ZSA+16] Man Zhang, Bran Selic, Shaukat Ali, Tao Yue, Oscar Okariz, and Roland Norgren. Understanding uncertainty in cyber-physical systems: A conceptual model. In *Modelling Foundations and Applications - 12th European Conference, ECMFA 2016, Held as Part of STAF 2016, Vienna, Austria, July 6-7, 2016, Proceedings*, pages 247–264, 2016.

PART VI

APPENDIX

"be ALWAYS honest with yourself and your team. The rest usually comes together."
ALICIA ASÍN PÉREZ

Appendix A

Industrial Domain Workflows

A.1 Automated Warehouse Workflow

A typical workflow of an Automated Warehouse (AW) is divided into two main groups as shown in Figure A.1, i.e., Warehouse Management System (WMS) and the Material Flow Controller (MFC). The WMS is the responsible to manage which orders have to be carried out. Instead, the MFC is the responsible for carrying out such orders. To store or retrieve goods, the WMS generates an order composed of multiple suborders. The goods Scheduling is able to manage all the generated orders and the suborders scheduling is the responsible of managing all the suborder that correspond to the same order. Once all the orders and suborders are scheduled, those need to be carried out. Therefore those suborders are transferred to the MFC. Note that a set of movements is required to complete a suborder. Each movement contains a set of actions including the information about which devices will execute them. The overall plan to execute the orders takes into account the orders' priority and the parallelization of the movements in an optimal way regarding time.

The MovementControl module is responsible for controlling and managing all movements which compose a suborder and fulfills the whole order. Note that a suborder is composed of different movements to be made by different devices in the form of actions. Thus, the MovementControl module dispatches actions to appropriate devices that carry out the designated actions. When the set of movements of the suborder is completed, the MovementControl will either terminate the order or the suborder. If only the suborder is finished, but not the whole order, the MovementControl will wait until the whole order is finished. Notice that MovementControl is informed when all the movements of the suborder are finished, that a suborder has finished. Note that goods need to go through different processes (e.g., weighing or labeling) that are carried out by different devices such as loading gauges and labelers. Also, note that

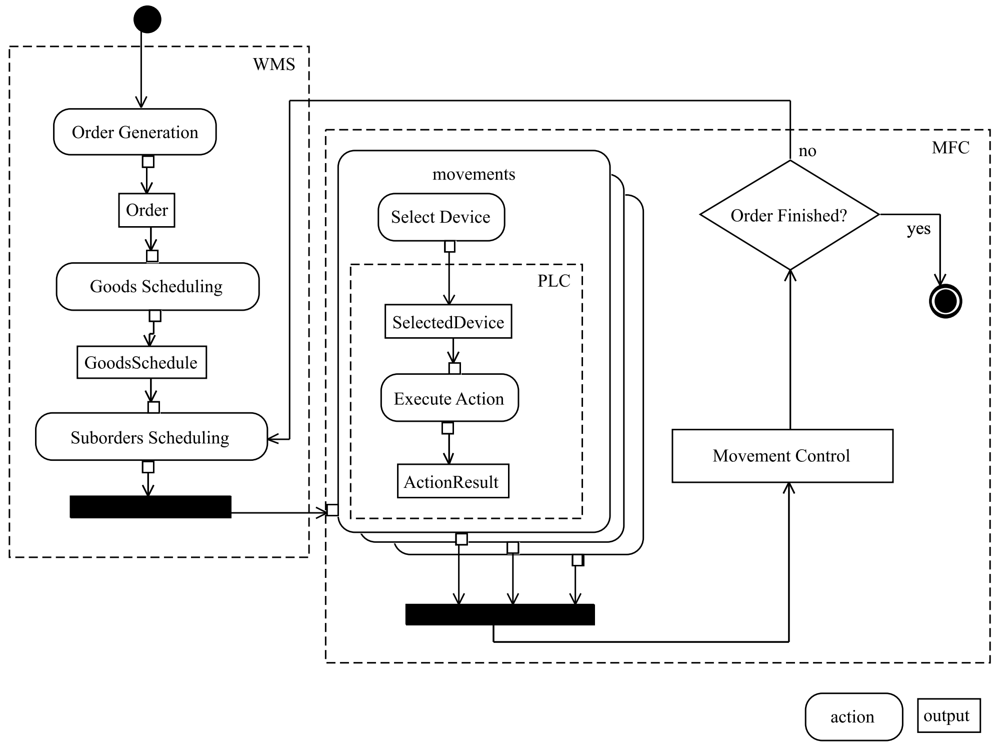


Figure A.1: Workflow of an Automated Warehouse

the activity diagram in Figure A.1 represents the process for storage or retrieval of single goods, but in a real scenario, multiple goods are managed at the same time. Parallelizing these tasks is complex due to the usage of shared resources (e.g., devices) and the problems related to it, such as deadlocks.

A.2 Press Machine Workflow

Figure A.2 shows the overall automated working process of a press machine for the production of a final work corresponding to a workpiece. To obtain the final work, a workpiece needs to be introduced in the press machine between the upper and lower dies. To shape a workpiece into a specific shape, it is necessary to attach the corresponding dies. The upper die is located under the ram, and the lower die is located above the bed as it can be seen in Figure 3.4Left. A press machine must be configured corresponding to the attached dies, in addition to many other components, and validate it before it can be used to carry out the final work. For example, one of the most important components is clutch-brake, which controls the speed of ram movements. The speed of the ram changes depending on the position. The speed is relatively fast at the beginning of the movement and is getting slower when approaching the bed. The speed of the ram should be properly configured since a higher speed contributes to a higher efficiency, i.e., producing final work more quickly, but at the same time increases the risk of breaking the dies, or other components. Depending on the workpiece and shape to be produced, the speed of the press machine must be configured to obtain the correct final work.

Another important factor that must be considered is the alignment of dies [LH12]. If the dies are not aligned properly, neither the upper nor the lower die fit together. Alternatively, the dies could fit together, but the pressure at all points could not be equal. Both problems will result in an imperfect final workpiece that would be unusable for its intended purpose. Apart from speed and alignment, there are several other configurations related to pressure, oil state, and temperature that are interesting for customers. For example, a customer can be interested in controlling the temperature, when a press machine is installed in an extremely warm environment requiring the machine to be stopped several times due to high temperatures. Another customer may be interested in controlling the oil contamination if such situation occurs in the customer's deployment. Thus, depending on customers' needs, a press machine will be composed of different physical elements, and their parameters must be configured. Figure 3.4Right illustrates some sensors that a typical press machine has such as, thermometers, pressure transducers, and flow meters. Note that, as mentioned previously, each customer has different requirements, thus, the number and types of devices vary.

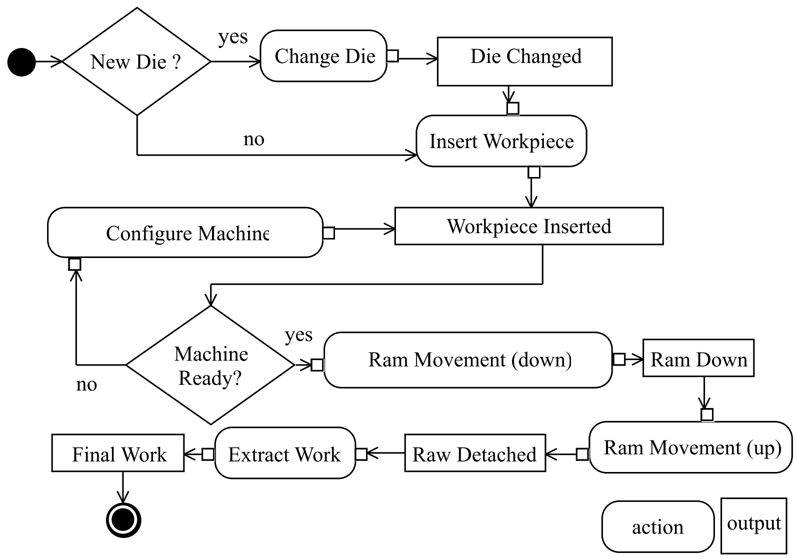


Figure A.2: Working Process of a Press Machine

Once all configurations (values assigned to configuration parameters) are validated, the ram starts moving down to press the introduced workpiece and complete the final work. Afterward, the platform returns to the initial safe position to take off the final workpiece. Notice that a wrong configuration can lead to imperfect final works, and during the production, a large number of final works are produced, then such imperfections can result in a significant monetary loss.

Appendix B

International Electrotechnical Commission (IEC) 61850 standards

The International Electrotechnical Commission (IEC) defined the IEC 61850 standard [TC-03] to model, control and monitor electrical substations. The standard defines a Basic Information Model, services to interact with it, and some recommendations for the use of different communication protocols. It is divided in different parts for general specifications, configuration, defining the model and communications, and testing.

IEC 61850 defines guidelines for monitoring and diagnosis of an electrical station. Although IEC 61850 is oriented towards electrical stations, its concepts still can be used in other industrial domains with a common goal to control and supervise devices of ICPSs [ICMU18a].

To model the intelligent electronic devices at electrical substations, IEC 61850 makes use of two building classes, i.e., Basic Information Model and Control Blocks (CB) for additional functions. The Basic Information Model defines the elements of the real world and defines their information with a simple structure:

- **Server**: exposes systems to the outside and includes one or more Logical Devices (*LDs*).
- **Logical Device**: virtual representation of a real device, composed of one or more Logical Nodes (*LNs*).
- **Logical Node**: virtual abstraction application functionalities. All *LDs* have a **Logical Node Zero (LLN0)** to represent common data for the *LD*.

- **Data**: physical world information, associated to a *LN*.
- **DataAttribute**: information piece of a *Data*, e.g., value, timestamp. The values of a *DataAttribute* are defined by a type (e.g., Float, boolean).
- **Dataset**: group of existing *Data* and *DataAttributes* of a *LD*.

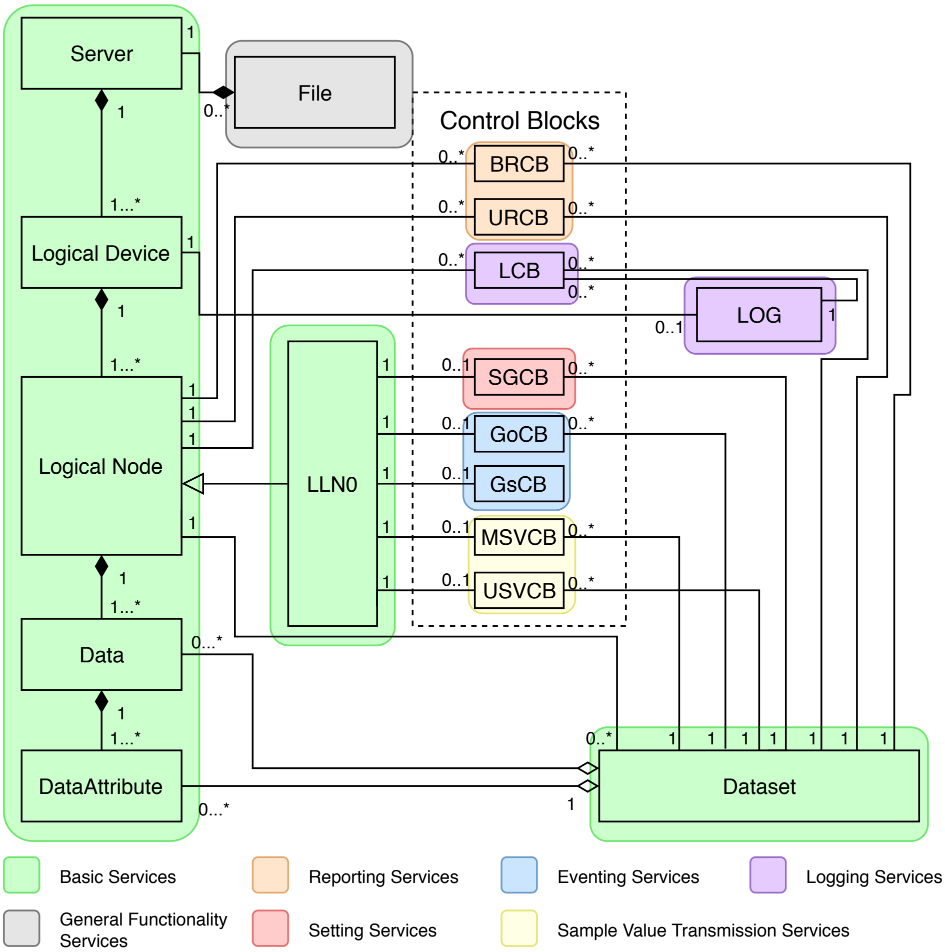


Figure B.1: The information model of the IEC 61850 standard.

The CBs are specialized classes to interact with the information model through some additional functionalities:

- **Reporting**: *Buffered Report Control Blocks (BRCP)* and *Unbuffered Report Control Blocks (URCB)* define the generation of reports, the former ensures that the reports arrive to the destination while the latter works on a best effort basis.

- **Logging**: *Log Control Block (LCB)* configures the creation of logs from Datasets, what to log and under which circumstances.

- **Configuration**: *Setting Group Control Block (SGCB)* groups settings and allows to change between the defined groups.

- **Eventing**: *Generic Object Oriented Substation Event (GOOSE)* and *Generic Substation State Event (GSSE)* are respectively managed by *GOOSE Control Block (GoCB)* and *GSSE Control Block (GsCB)* to deliver Datasets containing DataAttributes and basic state change information. The events are based on publish-subscribe communications.

- **Sampled Values**: manage the transfer of sampled information in Datasets of DataAttributes in a time controlled way. It can be implemented in two ways: using multicast communication with a *Multicast Sample Value Control block (MSVCB)* or unicast communication with an *Unicast Sample Value Control Block (USCVB)*.

Figure B.1 defines the different elements, both on the Basic Information Model and the CBs. All the elements have a name and an absolute reference to uniquely identify them throughout the entire model.

Appendix C

Questionnaire-based survey

C.1 Industrial Domain Information

Questionnaire: Monitoring Industrial Systems and Visualizing Information

Dear participant,

Thank you for sparing your valuable time to contribute to this survey. The objective of our study is to know more about your environment and have more information about future interest.

Some concepts that need to be clear for answering the questionnaire:

- Product: Every single project or element you create for your customers.
- Industrial System: An industrial product that is monitored. Ex. Automated Warehouse, Press Machine, Compressor, etc.
- Industrial Domain: A group of similar industrial systems. Ex. Automated Warehouses, Press Machines, etc.
- User Interface Component: Any component used for visualizing data. Ex. Pie chart, table, bar chart, etc.
- Device: Hardware that can be monitored, such as sensors, actuators, etc. A industrial system will be composed by different devices.

Conclusions from any survey are only as good as the data provided by the respondents. So, please answer all the questions accurately based on your personal knowledge and experience within the industrial setting, you are involved.

Confidentiality and Anonymity: This survey is NOT intended to assess individual knowledge and performance. Consequently, individual responses will be kept strictly confidential.

BACKGROUND

1. What is the size of your company?

○ 1-99 employees
○ 100-499 employees
○ 500-999 employees
○ 1000-2000 employees
○ More than 2000 employees

2. How would you assess your knowledge in SOFTWARE MONITORING SYSTEMS?

   (Question instructions: *Software Monitoring System: software capable of collecting device data.*)

○ Expert
○ Highly knowledgeable
○ Some experience
○ No experience

3. How long have you been working with SOFTWARE MONITORING SYSTEMS?

○ < 1 year
○ 1-4 years
○ 5-10 years
○ 10 years
○ Not important

4. What is your role within your company in terms of SOFTWARE MONITORING SYSTEMS?

☐ Developer
☐ User
☐ Designer
☐ Domain Expert
☐ Supervisor
☐ Manager
☐ Other: ____________

5. How would you assess your knowledge in DATA VISUALIZATION?

◯ Expert
◯ Highly knowledgeable
◯ Some experience
◯ No experience

6. How long have you been working with DATA VISUALIZATION?

◯ < 1 year
◯ 1-4 years
◯ 5-10 years
◯ 10 years
◯ Not important

7. What is your role within your company in terms of DATA VISUALIZATION?

☐ Developer
☐ User
☐ Designer
☐ Domain Expert
☐ Supervisor
☐ Manager
☐ Other: ____________

8. How many products are produced by your company every year?

- ◯ 1-9 products
- ◯ 10-49 products
- ◯ 50-70 products
- ◯ More than 70 products
- ◯ Don't know

PRODUCT CHARACTERISTICS

If you don't have information for answering, please try to give your opinion.

9. For your industrial systems. What kind of evolution can happen?

- ☐ Add new devices
- ☐ Remove devices
- ☐ Update devices
- ☐ Other: ____________

10. Do you think would be interesting to detect the evolution happened in your industrial system?

	Very interesting	Interesting	So-so interesting	Less interesting	Not interesting
In an automatic way	◯	◯	◯	◯	◯
at runtime	◯	◯	◯	◯	◯

11. If we would be able to detect the devices in an automatic way and at runtime in the industrial system...

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
the industrial system would improve	◯	◯	◯	◯	◯
it would be faster than configuring by hand	◯	◯	◯	◯	◯
would be less prone to errors	◯	◯	◯	◯	◯

12. How much would improve the industrial system, if we would be able to detect the devices in an automatic way and at runtime?

	1-20%	21-40%	41-60%	61-80%	81-100%	NOTHING
Due to time	◯	◯	◯	◯	◯	◯
Quantity of errors	◯	◯	◯	◯	◯	◯

13. How much would improve the industrial system, if we would be able to detect the devices in an automatic way and at runtime?

◯ 1-20% of the system
◯ 21-40% of the system
◯ 41-60% of the system
◯ 61-80% of the system
◯ 81-100% of the system

14. Would be possible to define the attributes of each kind of device before building the monitoring system?

◯ Yes
◯ No
◯ Depends: ____________

15. Every industrial system will be composed of different devices such as sensors, actuators, etc.

    VI: very interesting; I: interesting; SSI: so-so interesting; LI: less interesting; NI: not interesting

	VI	I	SSI	LI	NI
Would be interesting to detect commonalities and variabilities between devices of different industrial domains in order to reuse them?	○	○	○	○	○
If commonalities and variabilities exist, would be interesting to create a configurable software for configuring devices across the industrial domains?	○	○	○	○	○

16. If you share device configurations with other industrial domain, would . . .

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
be faster to start monitoring your industrial system?	○	○	○	○	○
improve the industrial system?	○	○	○	○	○

17. Is it important to display data coming from the devices?

    ○ Very Important
    ○ Important
    ○ So-so Important
    ○ Less Important
    ○ Not Important

18. If so, would be interesting to do it at runtime?

- ◯ Very Important
- ◯ Important
- ◯ So-so Important
- ◯ Less Important
- ◯ Not Important

19. If a software which leaves the user configuring a user interface component with the data coming from the devices exist. Would you apply it to an industrial system?

- ◯ Yes
- ◯ No
- ◯ Depends: ____________

20. If the user is able to configure the user interface component. Would the user be able to PROPOSE IMPROVEMENTS to the industrial system IN A FASTER WAY?

- ◯ Yes
- ◯ No
- ◯ Perhaps: ____________

21. If the user is able to configure the user interface component. Would the user be able to DETECT ANOMALIES in the industrial systems IN A FASTER WAY?

- ◯ Yes
- ◯ No
- ◯ Perhaps: ____________

22. If the user is able to configure the user interface component. Would the user HAVE A BETTER OVERVIEW of the industrial system?

- ◯ Yes
- ◯ No
- ◯ Perhaps: ____________

23. If we let the user configuring the user interface component at runtime...

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Do you think the user interface component need to have a special look and feel?	○	○	○	○	○
Do you think the style of different user interface components needs to be predefined?	○	○	○	○	○

24. If the system is able to analyze how the user visualizes the data...

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
the system will be able to propose better user interface components for visualizing data	○	○	○	○	○
the system will be able to help the user to configure in a better way the user interface component	○	○	○	○	○

25. Would be interesting to analyze how the user visualize the data?

- ◯ Very Interesting
- ◯ Interesting
- ◯ So-so Interesting
- ◯ Less Interesting
- ◯ Not Interesting

26. Make any comments you need. Give us your opinion.

◯ Comment: ________________________

PRODUCT CHARACTERISTICS

Try to classify industrial systems into three different categories: small, medium and large. Please answer the same questions for every category. If you don't have information for answering, please try to give your opinion.

27. By how many devices (actuators, sensors, displays) can an industrial system be composed?

	Less than 20 devices	20-49 devices	50-99 devices	100-500 devices	More than 500 devices	NOT KNOW
SMALL	◯	◯	◯	◯	◯	◯
MEDIUM	◯	◯	◯	◯	◯	◯
LARGE	◯	◯	◯	◯	◯	◯

28. Write comments regarding the quantity and type of devices.

Comments: ________________________

29. How many incidences can occur in an industrial domain during a week?

	1-9 incidences	10-29 incidences	30-50 incidences	More than 50 incidences	NOT KNOW
SMALL	○	○	○	○	○
MEDIUM	○	○	○	○	○
LARGE	○	○	○	○	○

30. Write extra comments about incidences.

    Comments: ____________________________

31. How much time (average), is needed for solving an incidence in an industrial domain?

	1/2 day	1-2 days	3-5 days	5-10 days	More than 10 days	NOT KNOW
SMALL	○	○	○	○	○	○
MEDIUM	○	○	○	○	○	○
LARGE	○	○	○	○	○	○

32. Write extra comments about solving an incidence

    Comments: ____________________________

33. How many people are supporting the software monitoring system?

	1-9 people	10-29 people	30-50 people	More than 50 people	NOT KNOW
SMALL	○	○	○	○	○
MEDIUM	○	○	○	○	○
LARGE	○	○	○	○	○

34. Write extra comments supporting the software monitoring system

    Comments: ____________________________

35. For each incidence caused in the industrial system, what is the probability that is due to a device?

	Less than 20%	20-49%	50-99%	100%	NOT KNOW
SMALL	◯	◯	◯	◯	◯
MEDIUM	◯	◯	◯	◯	◯
LARGE	◯	◯	◯	◯	◯

36. Write extra comments incidences caused by a device

Comments: ___________________________

37. When an incidence is caused by a device, how many times is it due to a physical problem?

	Less than 20%	20-49%	50-99%	100%	NOT KNOW
SMALL	◯	◯	◯	◯	◯
MEDIUM	◯	◯	◯	◯	◯
LARGE	◯	◯	◯	◯	◯

38. Write extra comments about physical problems in the devices

Comments: ___________________________

39. When a physical problem is found on a device, what is usually made? Add percentage to each case.

	SMALL	MEDIUM	LARGE
Remove and add a new one (%)	____	____	____
Fixed (%)	____	____	____
Remove (%)	____	____	____
Fixed (%)	____	____	____
Nothing (%)	____	____	____
Others (%)	____	____	____

40. Write extra comments about solving physical problems on the devices

Comments: ________________________

41. How often does a customer ask for industrial system improvement?

	< than 10 times every year	> than 10 times every year	Once every 5 year	Once every 10 year	Never	NOT KNOW
SMALL	○	○	○	○	○	○
MEDIUM	○	○	○	○	○	○
LARGE	○	○	○	○	○	○

42. When there is an improvement in an industrial system, in what percentage does this change affect the product?

	1-19%	20-39%	40-59%	60-79%	80-100%	NOT KNOW
SMALL	○	○	○	○	○	○
MEDIUM	○	○	○	○	○	○
LARGE	○	○	○	○	○	○

43. Write extra comments about industrial system improvements

Comments: ________________________

44. Make any comments you need. Give us your opinion.

Comments: ________________________

C.2 Modular Monitoring System

Modular monitoring System Evaluation

Dear participant,

Thank you for sparing your valuable time to contribute to this survey. The objective of our study is to know your opinion about the Modular monitoring System.

Conclusions from any survey are only as good as the data provided by the respondents. So, please answer all the questions accurately based on your personal knowledge and experience within the industrial setting, you are involved.

Confidentiality and Anonymity: This survey is NOT intended to assess individual knowledge and performance. Consequently, individual responses will be kept strictly confidential.

1. How many agents are usually reused from one warehouse to another?

   ◯ 1-24% agents
   ◯ 25-49% agents
   ◯ 50-99% agents
   ◯ 100% agents
   ◯ NONE

2. Additional comments about agent reuse

   Comments: ________________________

3. Development time.

	Less than 1 day	1-5 days	6-10 days	10-30 days	2-4 months	More than 4 months
How much is used for the supervisor in each installation?	○	○	○	○	○	○
How much is used due to new devices?	○	○	○	○	○	○

4. Additional comments about development

Comments: ________________________

5. Commissioning time.

	Less than 1 day	1-5 days	6-10 days	10-30 days	2-4 months	More than 4 months
How much is used for the supervisor in each installation?	○	○	○	○	○	○
How much is used due to new devices?	○	○	○	○	○	○

6. Additional comments about commissioning

Comments: ________________________

7. Make any comments you need. Give us your opinion.

Comments: ________________________

Appendix D

Scalability Test Results

D.1 Data Model Comparator Results

Table D.1: Obtained results from the data model comparator with **50 Logical Nodes**. Average time (AVG) and Confidence Interval (CI) of execution time needed (ms)

50 Devices								
	INSERT		MODIFY		REMOVE		RANDOM	
	AVG	CI ±	AVG	CI ±	AVG	CI ±	AVG	CI ±
20	18.06	0.773	16.288	0.602	14.462	0.443	16.155	0.627
40	19.553	0.508	16.347	0.457	13.062	0.478	15.403	0.525
60	21.026	0.589	16.183	0.556	11.649	0.228	18.28	0.747
80	22.654	1.086	16.127	0.45	10.06	0.309	14.265	0.763
100	23.795	0.734	16.281	0.518	-	-	16.73	0.502

Table D.2: Obtained results from the data model comparator with **100 Logical Nodes**. Average time (AVG) and Confidence Interval (CI) of execution time needed (ms)

100 Devices								
	INSERT		MODIFY		REMOVE		RANDOM	
	AVG	CI ±	AVG	CI ±	AVG	CI ±	AVG	CI ±
20	40.398	0.645	36.449	0.561	33.38	0.611	37.586	0.618
40	43.483	0.529	36.547	0.674	29.791	0.91	35.09	1.005
60	47.029	0.555	37.289	0.629	26.35	0.823	36.723	0.639
80	50.377	0.492	37.03	0.611	23.061	0.698	37.501	1.235
100	53.868	0.7	36.036	0.605	-	-	36.416	0.508

Table D.3: Obtained results from the data model comparator with **500 Logical Nodes**. Average time (AVG) and Confidence Interval (CI) of execution time needed (ms)

500 Devices								
	INSERT		MODIFY		REMOVE		RANDOM	
	AVG	CI ±	AVG	CI ±	AVG	CI ±	AVG	CI ±
20	179.135	2.836	159.188	2.234	144.423	1.828	161.05	2.366
40	198.863	3.486	164.873	3.699	127.603	1.774	161.808	2.217
60	214.576	3.57	161.443	2.364	114.108	1.576	159.476	2.093
80	233.163	4.82	160.746	2.112	94.59	1.258	154.081	1.793
100	258.216	4.841	162.846	2.117	-	-	36.612	1.965

Table D.4: Obtained results from the data model comparator with **1000 Logical Nodes**. Average time (AVG) and Confidence Interval (CI) of execution time needed (ms)

1000 Devices								
	INSERT		MODIFY		REMOVE		RANDOM	
	AVG	CI ±	AVG	CI ±	AVG	CI ±	AVG	CI ±
20	966.694	42.941	812.452	31.71	479.819	60.566	816.526	34.539
40	1062.361	41.325	805.063	34.992	390.658	39.605	781.418	23.924
60	1206.35	32.811	778.48	28.555	315.706	38.158	810.628	41.136
80	1312.73	56.793	812.255	37.455	226.409	4.901	759.616	16.87
100	1391.139	51.778	795.341	26.494	-	-	754.174	17.28

Table D.5: Obtained results from the data model comparator with **5000 Logical Nodes**. Average time (AVG) and Confidence Interval (CI) of execution time needed (ms)

5000 Devices								
	INSERT		MODIFY		REMOVE		RANDOM	
	AVG	CI ±	AVG	CI ±	AVG	CI ±	AVG	CI ±
20	7504.666	253.046	7948.106	192.028	4599.335	601.986	7948.296	208.028
40	8207.304	137.2	7983.115	199.595	3825.186	376.088	7871.655	223.714
60	7932.337	251.416	7937.625	195.618	3794.048	401.019	7123.38	570.081
80	7751.169	133.73	7906.775	186.239	3338.893	521.681	4872.885	694.389
100	10434.83	1422.032	8013.856	204.03	-	-	3743.969	259.904

Table D.6: Obtained results from the data model comparator with **10000 Logical Nodes**. Average time (AVG) and Confidence Interval (CI) of execution time needed (ms)

10000 Devices								
	INSERT		MODIFY		REMOVE		RANDOM	
	AVG	CI ±	AVG	CI ±	AVG	CI ±	AVG	CI ±
20	12723.95	383.693	12734.7	42.86	12254.94	33.845	12671.83	43.162
40	14616.28	82.877	12754.4	35.86	12168.72	402.94	12565.88	52.22
60	15310.77	97.376	12685.11	28.597	8709.968	363.877	12277.84	34.849
80	16461.61	85.731	12666.93	28.371	5620.353	52.583	11991.42	67.946
100	14637.95	439.539	12729.39	39.054	-	-	11779.71	27.591

D.2 Data Model Sizes

Table D.7: Data Model sizes in MB for comparison

Size	Qt-1 (MB)	Qt (MB)					
		Change	**20%**	**40%**	**60%**	**80%**	**100%**
50	1.515	Remove	1.212	911	0.609	0.307	–
		Insert	1.811	2.105	2.399	2.698	2.993
		Modify	1.513	1.506	1.500	1.496	1.494
		Random	1.453	1.328	1.862	1.082	1.558
100	3.024	Remove	2.420	1.818	1.212	609	–
		Insert	3.614	4.208	4.795	5.391	5.979
		Modify	3.017	3.007	3.000	2.990	3.026
		Random	3.136	2.655	3.035	3.057	2.930
500	15.106	Remove	12.083	9.066	6.043	3.024	–
		Insert	18.057	21.018	23.980	26.924	29.877
		Modify	15.065	15.022	14.974	14.928	14.888
		Random	14.891	14.592	14.692	13.246	14.385
1000	30.212	Remove	24.161	18.127	12.086	6.045	–
		Insert	36.119	42.028	47.941	53.837	59.751
		Modify	30.115	30.036	29.944	29.846	29.770
		Random	30.266	28.287	29.701	27.107	27.071
5000	147	Remove	117	88.4	59	29.5	–
		Insert	176	205	234	263	292
		Modify	147	146	146	145	145
		Random	143	136	126	112	99.6
10000	294	Remove	235	176	118	58.9	–
		Insert	294	411	469	295	585
		Modify	294	293	192	291	290
		Random	282	262	235	206	178