UNIVERSITÉ DU LUXEMBOURG

PhD-FSTM-2022-079

The Faculty of Science, Technology and Medecine

Dissertation

Presented on 01/07/22 in Luxembourg

to obtain the degree of

Docteur de l'Université du Luxembourg en Informatique

by

Julien Polge

Born on 2nd March 1994 in Nancy (France)

INVESTIGATING THE INDUSTRY 4.0 PARADIGM CHANGE IN BROWNFIELD MANUFACTURING FACILITIES: FROM DATA-DRIVEN APPROACHES TO BLOCKCHAIN-BASED TRACEABILITY

Dissertation defence committee

Dr. Jacques Klein, Chairman
Professor, University of Luxembourg, Luxembourg, Luxembourg

Dr. Eric Rondeau, Vice-Chairman
Professor, University of Lorraine, Nancy, France

Dr. Abdelaziz Bouras,
Professor, Qatar University, Doha, Qatar

Dr. Jérémy Robert,
Digital Technology Expert, Cebi Luxembourg S.A., Steinsel, Luxembourg

Dr. Yves Le Traon, Dissertation Supervisor
Professor, University of Luxembourg, Luxembourg, Luxembourg

Abstract

In the recent years, manufacturing companies have been transitioning towards the fourth industrial revolution (also referred to as Industry 4.0). One of the main objectives of this transition is for companies to increase their Overall Equipment Effectiveness (OEE), which is one of the most important production Key Performance Indicator (KPI). There are two main approaches to this transition, the first one is to start building facilities and infrastructures from scratch while directly having the Industry 4.0 features in mind (referred to as greenfield), the second one - *and the one we are focusing on in this dissertation* - is to retrofit the existing legacy machines and infrastructures, this is referred to as brownfield. The latter is also the preferred approach of Cebi Luxembourg S.A. - *our industrial partner in a Research, Development and Innovation project* - a company that has been existing for decades. In this context, this dissertation explores three aspects of the digital transformation, by investigating how innovative technologies/approaches can contribute to the establishment of secure, flexible and trustworthy solutions. This multifaceted work starts with the study of secure data collection within the factory, which is a fundamental requirement prior to any form of any further digitisation. The two complementary aspects of the thesis focus on different enabling technologies, namely Machine Learning (ML) to deploy data-driven and flexible process controls and blockchain as a secure and trustworthy mechanism to improve traceability in the production plant.

In the first part of the dissertation, we propose a security assessment of the Open Platform Communications - Unified Architecture (OPC-UA) protocol, from an application perspective. The objective of such a study is to verify whether the protocol proposes some security mechanisms and to what extent they could protect an application. This is of utmost importance for any manufacturing company, such as Cebi, in order to know whether they can rely on this protocol for their data collection infrastructure. For this purpose, we identify, based on the standard's specification, the threats and countermeasures that may be applied when implementing OPC-UA in an Industry 4.0 environment. In addition, we highlight the impact of two different attacks on an application. The results show that the

application can be disrupted and that the intrinsic security mechanisms absolutely have to be used, but additional security mechanisms have to be implemented at different layers in the data collection architecture.

In the second part of the dissertation, we investigate the integration of ML in the process control to increase the production flexibility. The intent of this study is twofold: i) overcome the problem of rigid controller programs that limit changes in the production lines and ii) investigate such an approach for manufacturing companies to be able to apply ML methods at the edge for various applications and to be aware of the implications of doing so. For this purpose, we propose a real case study based on a Time Series Classification (TSC) problem on a miniaturised factory controlled by true industrial Programmable Logic Controllers (PLCs) on which we step away from the paradigm of static controller programs. The results show that the ML method for TSC can be considered as a means to increase the production flexibility.

In the third part of the dissertation, we focus on traceability and blockchain for this purpose. A technological watch on blockchain technologies is essential for manufacturing companies, since the big manufacturing groups are more and more investigating these technologies for their supply chains, due to the immutability property provided (e.g., Cebi's clients are exploring blockchain solutions in the automotive industry). As a starting point, we identify and compare five of the major permissioned blockchain frameworks used in the industry, based on the literature. This study shows that two frameworks stand out, but the choice of the framework has to be a trade-off. Then, in another study, we implement a private Ethereum platform on our partner's servers, in order to verify the suitability of such a framework for traceability purposes. Finally, after identifying all the parameters and their influence on the outcomes in the previous study, we propose BlockPerf, a simulator/emulator framework designed to implement and to compare different blockchain platforms. At this stage, BlockPerf is compared to BlockSim (a major simulator in the literature) and to a real bitcoin blockchain implementation. Our study shows that our framework improves the results provided by BlockSim by 50% on average, while also implementing the dynamic nature of a blockchain network, and closer results to the ones obtained out of the bitcoin implementation.

The three contributions are representative of some of the encountered challenges when engaging a transition towards Industry 4.0 in brownfield facilities in a general manner.

Acknowledgments

I was lucky enough to receive a significant amount of support and assistance throughout this journey.

First, I would like to thank Pr. Yves Le Traon who allowed me to integrate his research group to pursue my PhD studies under his supervision. He believed in my work and provided me with valuable feedback and ideas. In addition, I am grateful to my co-supervisor and daily advisor Jérémy Robert for his support and patience. His implementation advice as well as his input on presenting my research work were greatly appreciated. The discussions we had allowed me to be more rigorous and to improve my research overall. I also want to thank the defence committee members for their interest in my research and the time they invested in my dissertation.

I am also thankful to my co-authors Dr. Sylvain Kubler and Sankalp Ghatpande for their contributions and feedback that helped reinforce this thesis.

I would like to show my gratitude to the Luxembourg National Research Fund (FNR) for their trust in the initial ideas that lead to this dissertation and for granting the funding that supported my thesis. Identically, I would like to thank Cebi Luxembourg S.A. and more specifically Guillaume Policarpo, leading the project at Cebi, for providing a real industrial playground with requirements and expectations that added different perspectives to my work.

I would like to thank also all my colleagues from SERVAL (SnT) for their feedback, expertise and help when needed.

Finally and on a more personal note, I would like to thank from the bottom of my heart my mother Laurence as well as my father Franck, who supported me since the very beginning and provided me with life advice when needed. I would also like to thank my brother Antoine for supporting me. I want to thank my colleague and friend Dr. Paul-Lou Benedick, with whom I shared the last nine years as a student and with whom I shared great discussions. Last but not least, my special thanks to my friends Valentin, Jordan, Steven and Thomas for their continuous support and the great times we had throughout this journey.

Contents

1

Introduction

Modern industry is shifting from the third industrial revolution, during which the processes' automation has been enabled by the introduction of computers and Programmable Logic Controllers (PLCs) into production plants, to the fourth industrial revolution - *also known as Industry 4.0* -. The term Industry 4.0 was originally introduced by the German government as *Industrie 4.0*, and was initially a proposal for developing a new concept of German economic policy in 2011 [RMK16]. Enabled by different technologies and concepts, such as the the development of the Internet of Things (IoT) in the Industry (Industrial Internet of Things (IIoT)), new decision-making processes integrating Machine Learning (ML) algorithms, and Cyber Physical Systems (CPS), the main features of the Industry 4.0 can then be summarised as the following [PTB+15, LFK+14, Jaz14, VHH16, BMR+20]:

- **Customisation/Individualisation of the products:** the production needs to adapt to the customer's requirements that tend to be increasingly precise and individual. This also allows for the development of innovative business models.
- **Production flexibility:** the production line needs to automatically adapt itself (self-configuration) to the evolving requirements (e.g., the range of products it should produce), to enable productivity gains while also playing an important role in the individualisation of the products.
- **Traceability:** one needs to be able to identify all the processes through which raw material, parts, and products went.
- **Optimisation of the production process:** thanks to the IoT devices and the data gathered and analysed by ML algorithms, it is possible to have a precise overview of the production process and, for example, optimise resources or predict forthcoming maintenance actions.

One of the main goals to achieve for manufacturing companies willing to step into this fourth industrial revolution is to adopt the aforementioned features while increasing the overall production performance. This is commonly measured using the Overall Equipment Effectiveness (OEE), which is an important production Key Performance Indicator (KPI) in the manufacturing industry.

Generally, there are two approaches to engage the transition towards Industry 4.0. The first one, which is referred to as *greenfield*, consists in (re)designing the production plants as well as the infrastructures from scratch such that they natively integrate Industry 4.0 features. The second one is referred to as *brownfield*, and consists in retrofitting the existing machines and infrastructures such that they can support the Industry 4.0 features. The latter approach is often the one preferred by manufacturing companies that have been established for decades, especially since their premises are composed of hundredth of legacy machines. It would cost a lot to replace every machine and the whole infrastructure that supports these machines. However, it is difficult for companies to start such a transition since it

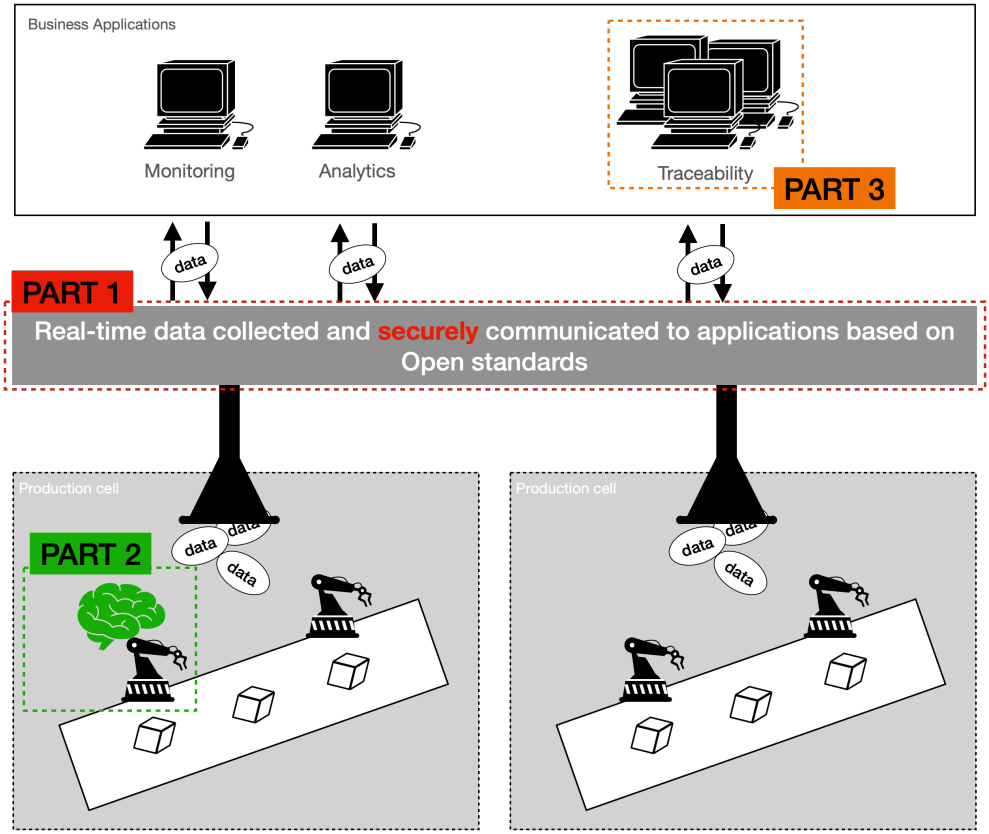


Figure 1.1: Envisioned Industry 4.0 infrastructure

implies considerable changes in terms of technologies to be implemented, for which they might not have the expertise in-house, but also cultural and organisational changes. As a matter of fact, in order to integrate the Industry 4.0 enablers in a *brownfield* context, one of the first step is to collect data. Indeed, a secure data collection architecture lays the foundation for any data-driven approach enabled by production data being collected from production cells, stored, analysed and processed by business applications as depicted in Figure 1.1. These data-driven approaches are the most significant part of the shift towards Industry 4.0, since during the previous industrial revolutions, decisions were taken based only on human experience and not on the important volume of data collected thanks to the advent of IIoT. Consequently, it is required to interconnect the Operational Technologies (OT) network - *composed of legacy equipment, often subject to vulnerabilities* - to the Information Technologies (IT) network - *which encompasses critical business activities* -. This must be done in an interoperable, standardised and most importantly secure manner since a vulnerability in a legacy OT device then becomes

a vulnerability for the whole network after interconnection. Therefore, this interconnection can create some misgivings and the risks have to be analysed, security measures have to be implemented and the extent to which these measures mitigate the risks has to be assessed.

Then, data being collected and accessible from anywhere on the network infrastructure, thanks to the interconnection framework, allow for the development of Artificial Intelligence (AI) applications in order to reach the ultimate goal of 'Self-X' properties, e.g., Self-configuration, self-repair, and so forth. But at a production cell level and from a process optimisation perspective, the question arises of the extent to which this AI could be used in the process automation architecture and whether it can improve the flexibility of the process. Indeed, to increase the production flexibility as well as to enhance the products individualisation possibilities, it is required to step away from the static PLC programs that are not meant to evolve and to adapt to a changing environment, and consequently reconsider the way the process is controlled. Technologies used to replace the static programs have to be proven at least as effective as the programs prior to any implementation in production settings, so as to dispel any reluctance.

The data collection infrastructure also allows for precise traceability, whether it is product or process information, including potential ML decisions that can also be collected through the interconnection framework. However, establishing a trustworthy product traceability system is a challenging and expensive task, for which it is difficult to measure the return on investment [MSS18]. But for trust and security reasons, it is required to move away from traditional centralised traceability systems, since they are exposed to data tempering as well as being a single point of failure [XLL+19]. Storing data in an immutable manner is an important asset, especially when different writers/groups/companies are able to access the system. This requires designing the traceability system while keeping in mind the weaknesses of the traditional centralised systems and the opening up to external or new participants.

To support companies in this transition, it is common to witness the creation of partnerships between industrial and academic actors. Such partnerships allow industrial companies to get advice and expertise from the academic actors and they allow the academic actors to dispose of actual industrial inputs and use cases. In this context, Cebi Luxembourg S.A. - *an industrial company specialised in the production of parts such as temperature sensors, oil and water level sensors for the household appliances industry and the automotive sector* - and the Interdisciplinary Centre for Security, Reliability and Trust (SnT) partnered through the two following projects:

- a Research, Development and Innovation project started in 2017 and supported by the ministry of economy of Luxembourg whose ultimate objective

is to upgrade Cebi's on-site infrastructures in order to meet the Industry 4.0 requirements in terms of key technical enablers; e.g., integrate a live data analytics platform that can compare live and historical data so as to detect performance deviations and anomalies before they become significantly impactful, or implement a process supervision system to improve the process at runtime. This project has a process-oriented focus.

- a BRIDGES project started in 2019 and supported by the Luxembourg National Research Fund which aims at i) designing a flexible infrastructure ensuring trust and immutability of product-related information and ii) investigating the impact of potential artefacts in the traceability and analysis processes. This project focuses more on product-related information.

In that context, this thesis explores three aspects of the transition towards Industry 4.0 in *brownfield* manufacturing facilities, by investigating how innovative technologies/approaches can contribute to the establishment of secure, flexible and trustworthy solutions. Figure 1.1 presents the envisioned infrastructure we are investigating in this dissertation.

The first part of this dissertation focuses on the security aspect of the data exchange inside of the factory (referred to as Part 1 in red colour in Figure 1.1). For that matter, we propose to assess the security mechanisms of the Open Platform Communication - Unified Architecture framework which is becoming the *de facto* standard in the industry. The objective is to identify the potential threats when using this protocol as well as the countermeasures implemented by the protocol. Additionally, we conduct experiments to assess the impact of some attacks from the application perspective. This allows manufacturing companies, such as Cebi, to decide whether they can rely on this protocol for their data collection architecture.

The second part of this thesis is oriented towards the production flexibility feature (referred to as Part 2 in green colour in Figure 1.1). In this part, we explore to what extent the integration of ML in the process control can improve the flexibility of the production. In addition, it can be considered as a feasibility study for manufacturing companies to be able to apply ML methods at the edge for various applications and to be aware of the implications of doing so. We conduct experiments based on a Time Series Classification (TSC) case study on a scale model of a factory in which the process is controlled by actual industrial PLCs.

The third and last part of the dissertation explores the use of blockchain for traceability purposes (referred to as Part 3 in orange colour in Figure 1.1). Big manufacturing groups are taking part in various blockchain initiatives, and start to use such technologies for their supply chain (e.g., Cebi's client are exploring blockchain solutions in the automotive sector). Manufacturing companies therefore need to conduct a technological watch on blockchain technologies. For that matter, in this part, we identify and compare five of the major permissioned blockchain

frameworks used in the industry, based on the literature. Then, we propose a methodology for selecting an optimal input configuration and apply it using a private Ethereum platform implemented on our partner's servers. Finally, we propose BlockPerf, a simulator/emulator framework designed to implement and to compare different blockchain platforms and evaluate it against BlockSim (a major simulator in the literature that BlockPerf actually extends) and against a real bitcoin blockchain implementation. BlockPerf is available to the community.

This thesis is a compilation, i.e., each chapter is a peer-reviewed and published paper except Chapter 5. Contexts and additional information are given as forewords at the beginning of each part (in red boxes) and high level conclusions are also given after some chapters (in green boxes).

The dissertation is organised in three parts, Part I focuses on the security aspect of the data collection architecture and is composed of Chapter 2 in which we propose a security assessment of OPC-UA from an application perspective. Part II refers to the flexibility of the process control and is composed of Chapter 3 in which we propose to integrate ML in the control architecture. Finally, Part III investigates the use of blockchain for a trustworthy traceability system and is composed of three chapters: Chapter 4 proposes a comparison of five of the major permissioned blockchain platform based on the literature, Chapter 5 proposes a methodology for finding and selecting an optimal blockchain configuration and Chapter 6 proposes BlockPerf, a blockchain simulation/emulation framework. Chapter 7 concludes the dissertation.

Part I

Secure IT/OT interconnection infrastructure

Foreword

Engaging the transition towards Industry 4.0 in a *brownfield* environment requires the interconnection of the existing OT infrastructure to the IT infrastructure to access data at any level of the global infrastructure. This data collection infrastructure is the foundation of any other data-driven approach. However, in most cases the OT infrastructure is composed of legacy components. The main issue with these legacy components is that they lack of security by design. Indeed, these components are often resource-constrained and have been developed years/decades ago. Consequently, connecting these components to the IT infrastructure that supports (critical) business applications can be risky since the vulnerabilities of the legacy OT components will be added to those of the IT infrastructure and the attack surface will be increased. This leads the following question:

Is it possible to create a secure interconnection infrastructure composed of legacy vulnerable equipment?

In this context, there are several considerations: i) the OT and IT assets and their vulnerabilities and ii) the security mechanisms proposed by the interconnection framework.

Regarding the first consideration, it is important to conduct an inventory of the OT and IT assets as well as an identification of their vulnerabilities before any interconnection, to be aware of all the potential risks and their criticality and then conduct a risk assessment study. This is necessary for envisioning the countermeasures to implement and the mitigating actions needed. For this matter, a risk analysis based on the ISO 27005 can be done. This analysis starts by an identification of the assets - *whether these assets are business processes and activities or hardware, software, networks, personnel and so on* -. Then the assets have to be valuated using two criteria: the replacement value of the asset and the business consequences of a loss or a compromise of the asset. The second step consists of identifying the threats and their source. Afterward, the third step is the identification of the controls (countermeasures) and the vulnerabilities that can be exploited by the previously identified threats. After this, the consequences have to be identified. Subsequently, the risk analysis phase can be conducted. This phase is composed of three sub-steps, namely, consequences assessment, incident likelihood assessment and level of risk determination. Finally, the risks can be evaluated by comparing the risk levels against the risk evaluation criteria and the risk acceptance criteria. The four remaining steps are the following: Information security risk treatment, Information security risk acceptance, Information security risk communication and consultation and

Information security risk monitoring and review. This study is a prerequisite to any further interconnection since it allows for an identification of all the assets and their vulnerabilities and it contributes to a better knowledge on the current system and the system to be. We supported Cebi in this risk assessment study. However, since it does not bring any scientific problem, this will not be further detailed in this manuscript.
For the second consideration, we are focusing on the OPC-UA framework. OPC-UA is becoming the *de facto* standard in the industry and is recommended by the German initiative Industrie 4.0 for unifying the industrial communications. In other words, this framework is the one used for accessing data produced in the production plant from the business IT network in a secure and interoperable way, regardless of the underlying protocols, standards and brands of controllers. The intrinsic security mechanisms therefore need to be evaluated prior to the adoption of this framework for such a strategic use. For this matter, we identify, based on the standard's specification and the understanding of the system architecture revealed by the risk assessment study, the threats and countermeasures that may be applied when implementing OPC-UA in an Industry 4.0 environment. In addition, we highlight the impact of two different attacks from an application perspective.
This chapter presents our study published and presented at a peer-reviewed conference - *IEEE Annual Consumer Communications & Networking Conference (CCNC)* - in 2019 and entitled *"Assessing the impact of attacks on opc-ua applications in the industry 4.0 era"* [PRLT19].

2

Assessing the impact of attacks on OPC-UA applications in the Industry 4.0 era

Contents

2.1 Introduction

Over the past few years, digital revolution has empowered our world: from the personal life to businesses. Indeed, the information is available anywhere, anytime, with any content for any user, which owns a device with an Internet connection. All of that was enabled by the advent of the IoT. This concept has raised many 'Smart City' initiatives all over the world [NDMC+14]. Most of them aim to create a "System-of-Systems" [NBH+11], where disparate information systems, sensors, devices, people and software solutions are used altogether.

The trend is to apply those concepts to the industrial world. Different consortia, such as Platform Industrie 4.0 or Industrial Internet Consortium (IIC), are therefore been created for bringing the Internet of Things into the industry (also referred as Industrial Internet of Things). It becomes a flourishing research topic, motivated by the possibility to develop new business models for manufacturers and their customers, while enabling productivity gains. One of the most important issues is to break down the vertical silos that shape today's industries (as in IoT applications). Most of the companies still have heterogenous, non-interoperable and proprietary systems at the shop floor, which are not yet connected to the IT network.

In this context, the German initiative Industrie 4.0 recommends the OPC-UA framework so as to unify the industrial communications. The Open Platform Communications (OPC) Foundation is in charge of managing a global organisation in which users, vendors and consortia collaborate to create the transfer of this standard for multi-vendor, multi-platform, secure and reliable interoperability. This is all the more important as industrial automation systems need to evolve, from a static pyramidal structure in which there was a separation between the OT and the IT networks to a dynamic and flexible flattened infrastructure, enabling to easily collect data from the shop floor to the business systems. This flexibility is indeed more and more important considering the need for strong product customisation ("on-demand" production).

However, those flattened architecture needs to perform a high-level security. Within this OT/IT convergence context, it is even more important to secure the network and IoT ressources, that could be used by malicious users to gain access to Industrial Control Systems (ICS) or Supervisory Control and Data Acquisition (SCADA) systems. Integrating new protocols in the product lifecycle of manufacturing systems can therefore pose several security challenges [CRFAF17]. Indeed, all these systems brought together in a heterogeneous environment may lead to create a new attack surface, that is different from the traditional industrial security issues [vV17]. In addition, IIoT networks are more and more targeted by cyber-attacks, in particular because of the lack of security of the IoT devices [AAKS17]. This is mostly due to the low resource constraints that come with these devices,

and they are particularly attractive because they often exchange sensitive and critical data through poorly secured networks [FBVI17].

The contribution of this paper is twofold: i) identifying, based on the standard specifications, the threats and countermeasures/mitigations that may occur/be applied when using OPC-UA in an Industry 4.0 environment and ii) highlighting the impact of the eavesdropping and message flooding attacks on an OPC-UA application implemented on a real testbed.

The rest of the paper is organised as follows: Section 2.2 presents the OPC-UA architecture with a particular focus on the security aspect. Section 2.3 aims at modelling the OPC-UA threats and the known countermeasures in an Industry 4.0 environment. Different experimentations have been led in Section 2.4 for highlighting the impact of the two aforementioned attacks on an OPC-UA applications. Section 2.5 discusses on new standards that can be used for reinforcing the security. Finally, Section 2.6 concludes this paper.

2.2 OPC-UA & Security

Originally, OPC-UA comes from the automation world with the “classic" OPC (OLE for Process Control) protocol [Fou02], whose it is the successor. Developed by the OPC Foundation, it is standardised under the IEC-62541 identifier [Com16], and consists of 14 parts (almost in release 1.04). OPC-UA is intended to be used as Machine-to-Machine (M2M) architecture enabling to interconnect sensors, field devices, PLCs, and applications at the shop-floor. It can also be implemented between devices at the shop floor level and various kind of systems, such as SCADA systems or Manufacturing Execution Systems at the enterprise level. OPC-UA, as an application protocol, relies on a clients/server protocol. The OPC server acts as an IIoT gateway (or wrapper in brownfield installations) exposing information about the system, its configuration topology and data context through a specific format (referred as address space), that can be discovered and accessed by OPC-UA clients. OPC-UA is implemented on top of the Transmission Control Protocol (TCP)/Internet Protocol (IP) stack.

As our focus is the security of the OPC-UA protocol, the part 2 of the specification is of utmost importance. This part defines six security objectives: i) *Confidentiality*, ii) *Integrity*, iii) *Availability*, iv) *Authentication*, v) *Authorisation* and vi) *Auditability*. According to the Federal Office for Information Security, the objective of *Non-repudiation* needs to be added to the specification and tackled by the protocol [fISB17]. The security is set up at the application layer, and is managed by two different sub-layers called ‘Application’ and ‘Communication’. The ‘Communication’ sub-layer is responsible for establishing an OPC secure channel between the client and the server. This secure channel guarantees *Confidentiality* by encrypting data, *Integrity* by digitally signing messages, and *Application*

Authentication with the use of digital certificates. When a secure channel is establishing, client and server need to agree on a common algorithm (for signing and encrypting messages but also for key derivation) referred as *Security Policies* in OPC-UA terminology. The security policies supported by the server will be announced *a priori* so as to give the choice to the client. The available security policies (version 1.03) are: *none, Aes128-Sha256-RsaOaep, Basic256Sha256* and *Aes256-Sha256-RsaPss*. After that, an OPC session is then established (at the 'Application' sub-layer) over the secure channel and relies upon the latter for secure communications. This session aims at ensuring *User Authentication* by using user credentials (i.e. name and password) and *User Authorisation* with permissions based on basic user roles (e.g. read and write).

Looking at the literature, few work has been led on the security of OPC-UA. The aforementioned mechanisms (OPC secure channel and session) have already been analysed in a formal way from a cryptographic perspective (by using a formal verification tool like ProVerif) [PPL16]. As digital certificates are used by the OPC-UA security mechanisms, some authors pointed out that the management (such as the distribution, the validation or the revocation) of those certificates is important and for that reason they propose an overview of suitable frameworks and architecture for this purpose [FK12]. Let us note that these studies rely only on theoretical models or subjective assessment of frameworks, and not on real implementation. Regarding real implementation assessment, the study described in [CCM10] aims at evaluating the impact of the different security mechanisms on the time to establish a connection between the client and the server. To put it in another way, no work focused on identifying the main potential attacks that can occur in an Industry 4.0 environment when using OPC-UA, and on highlighting the impact of some attacks on the application from an experimental perspective.

2.3 OPC-UA Threat Model

The proposed OPC-UA Threat Model is divided into 3 different parts: i) the threat sources (cf. 2.3.1), ii) the threats, impacts and countermeasures (cf. 2.3.2) and iii) its application in an Industry 4.0 environment (cf. 2.3.3).

2.3.1 Threat Sources

The potential threat sources to an OPC-UA system identified are similar to those pointed out in [AM14, FBVI17]:

1. **Internal user:** This is the owner of the device or a user having legal access to it. This user can uncover flaws in the system, launch attacks from it and is able to retrieve confidential information to sell it to a third party, or to attack another similar system. An internal user can be malicious or careless.

2. **Lack of Security-by-design:** This is when a manufacturer (deliberately or not) introduces backdoors in the devices he produces. Then he (or any other attacker since there are backdoors) is able to exploit the breaches to gain information about the users, the network topology and the devices connected to it, or simply gain remote access to the device.

3. **External attacker:** This is an attacker that is outside of the system, which has no authorised access to the system. It could be amateur or professional hackers, rival corporations, etc. This attacker will try to gain access to the system to retrieve sensitive information, to bring down services, to cause malfunction to the system, etc.

Table 2.1: Security objectives impacted by threats

Type \ Objective	Confidentiality	Integrity	Availability	Authentication	Authorisation	Auditability	Non-repudiation
Direct	*1, 2, 3, 4*	*2, 5, 6, 7*	*2, 3, 7, 8*	*2, 3*	*2, 3, 4, 5, 6, 7, 9*	*2, 3*	*10*
Indirect	*11*						

2.3.2 Threats, Impacts and Countermeasures

The standard partly lists the following threats: *1.* Eavesdropping, *2.* Session Hijacking, *3.* Rogue Server, *4.* Compromising User Credentials, *5.* Message Spoofing, *6.* Message Alteration, *7.* Malformed Messages, *8.* Message flooding, *9.* Message Replay, *10.* Repudiation and *11.* Server Profiling.

The security objectives - *impacted by those threats* - are gathered in Table 2.1.

2.3.3 OPC-UA Threat Model in an Industry 4.0 environment

Figure 2.1 depicts an example of industrial/manufacturing environment (referred here as Industry 4.0), from the outside world to the shop floor. There are 3 production cells/units, in which operate a robot (or many) and several distributed Inputs/Outputs (I/O) (sensors and actuators) managed by a PLC. Those PLCs, that do not have OPC-UA capabilities (i.e. in Production Cell 1 and 2), communicate with the distributed I/O by legacy fieldbuses (e.g. DeviceNet, Profibus, ...). An OPC-UA gateway is therefore responsible for exposing data coming from the PLC (and its distributed I/O) into an OPC-UA address space. Let us note that in this exemple, the PLC with OPC-UA capabilities is directly connected to a real-time (switched) Ethernet network. Those PLC and OPC-UA gateways are connected to the shop floor network (e.g. through a switched Ethernet network) where one can find aggregated OPC-UA servers but also various Human Machine Interfaces (HMI), cameras, cell performance displays and so on (not all represented

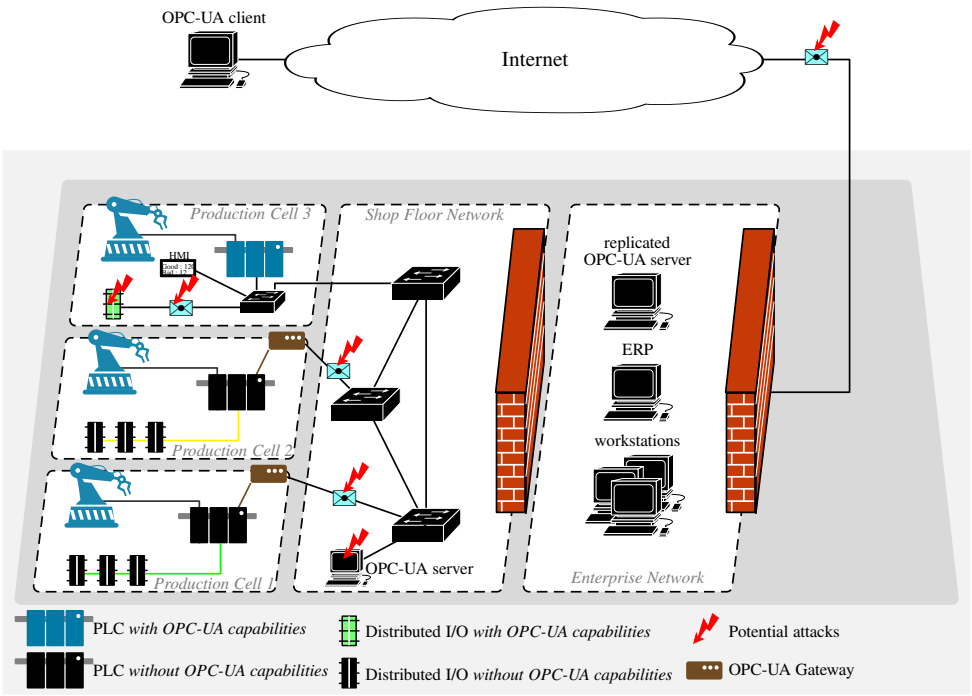


Figure 2.1: OPC-UA Threat Model in an Industry 4.0 environment

in this figure). Although the architecture is a flattened infrastructure due to the use of Ethernet from (almost) end-to-end, it is nonetheless divided into different "levels" for security purposes. Indeed, firewalls will still be used between the shop floor network and the entreprise network, and between this latter and the outside world. In addition, a replicated OPC-UA server can be used at the enterprise network so as not to disturb the shop floor (and limit potential attacks) but also for selecting the data that can be exposed to the outside world (e.g. for protecting the intellectual property of the process).

Now, looking at the location of the aforementioned attacks, *Eavesdropping*, *Server Profiling*, *Message Spoofing*, *Malformed Messages*, *Compromising User Credentials* and *Message Alteration* can occur everywhere as long as OPC UA is used (e.g. as depicted in Figure 2.1, between the OPC-UA gateway and the OPC-UA server at the shop floor level, or between PLC and distributed I/O in the production cell 3 or even between an outside OPC-UA client and the replicated OPC-UA server). *Message Flooding* could happen in the enterprise network if the attacker is outside of the facility, but it could happen on the production cells or in the shop floor network if the attacker is inside. This attack can lead to a deny of service either on the OPC-UA server, gateways or even on the distributed I/O. *Session*

Hijacking could occur anywhere an OPC UA session is opened. An attacker could also create a spoofed OPC-UA Client and impersonate any of the remote clients to retrieve the information exposed in the replicated OPC UA Server.

Let us note that despite the countermeasures implemented by OPC-UA - *that rely on relatively generic and well-known mechanisms for this kind of attacks* -, it is needed to implement other and external countermeasures at different levels. Indeed, the attacks do not target only the OPC-UA protocol, but also the other protocols of the stack (e.g. TCP, HTTP, *etc.*). And sometimes, these other protocols (in particular TCP) are used for creating an attack such as *Message Flooding*, that obviously can take the OPC-UA services down. That is one of the reason why firewalls have still been depicted in Figure 2.1 that have to implement mechanisms as Deep Packet Inspection (DPI) for detecting the attacks before taking actions. To do so, it is needed to have the knowledge of the impact of such attacks on the OPC-UA application itself. The next section gives an overview of this impact through several experiments.

2.4 Experimental results

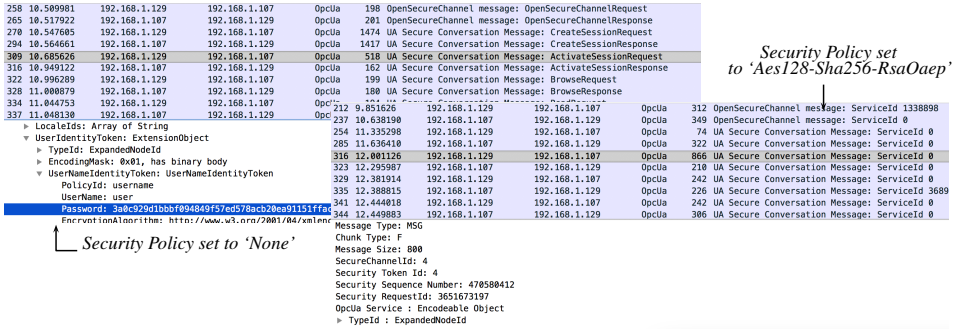


Figure 2.2: Wireshark capture with Security Policy set to 'None' and to 'Aes128-Sha256-RsaOaep'

The objective is to experiment a scenario that could really be implemented inside a production cell, and in particular the third one in the Figure 2.1. To do so, an OPC-UA server and an OPC-UA client (acting as the PLC and the HMI) have been implemented. Both of them are running on Raspberry PI 3 (with a Quad Core $1.2GHz$ 64bit Central Processing Unit (CPU) and $1GB$ RAM) that are connected to a switched Ethernet network. Amongst several free-of-charge implementations available (such as Open62541, FreeOpcUa, node-opcua, Eclipse Milo or opcua4j), Eclipse Milo has been preferred. As a matter of fact, as stated in [HMM+], it is a very complete solution implementing all the security mechanisms specified in the standard and enabling to develop client *and* server. In addition,

it is a Java implementation and can therefore be run on any operating systems (including Linux as on our Raspberry PI). In our application/development, the OPC-UA server exposes 78 information related to the production efficiency (e.g. the number of pieces correctly manufactured or not correctly produced and the process cycle time) of this production cell. An HMI displays those information by requesting them every 10 ms through an OPC-UA client (that is implemented on one of both Raspberry PI). The native/binary profile has been used in this application that is running on the TCP port 12686 on the server side. Let us assume that an (internal) attacker has been able to connect (or used a device already connected) to the network of the production cell 3. This attacker performs two different attacks: i) first, a man-in-the-middle attack to try to get (or steal) user information (like username and password) and ii) a message flooding attack to disturb the current OPC-UA application (that is only a monitoring one, but could be a critical - *and time-constrained* - application). For the sake of the experiment, a MAC book pro (with a Quad Core 2.8GHz CPU and 16GB RAM) has been used as attacker's laptop, where different tools are available (e.g. Wireshark, nmap, hping3). Next Subsections present results and findings related to those two attacks.

2.4.1 Man-in-the-Middle attack & OPC-UA security policies

Once connected to the network, the attacker (performing a man-in-the-middle attack) replies to all the ARP requests, by indicating that the server's (and client's) IP is at his/her own MAC address (well-known ARP Spoofing technique). Let us note that either the attacker has already the knowledge of the IP addresses or he/she was able to discover them by using a tool like *nmap*. The ARP spoofing is possible due to the dynamic ARP table on the devices (and in particular on the OPC-UA client). After having successfully falsified the ARP table of the server and the client, he/she received all the traffic between them on his/her computer (i.e. this traffic pass through his/her computer to be more precise). The attacker is therefore eavesdropping the communications by capturing them with a tool like TCPdump/Wireshark.

In this experiment, the security policies of the OPC-UA application has been set up either to 'None' or to 'Aes128-Sha256-RsaOaep'. Figure 2.2 shows an excerpt from the traffic trace the attacker has been able to capture. The first capture (with security policy set to 'None') highlights the behaviour of the two OPC-UA sub-layers 'Application' and 'Communication' that respectively create/open a secure channel and a session (as explained in the Section 2.2). Let us remind that this latter sub-layer is responsible for managing authorisation and authentication. The session opening request frame (underlined in the Figure 2.2, and numbered 309) is therefore interesting to look at. Indeed, the "UserNameIdentityToken" field give

access to the user information: for instance, in this trace the policy used is based on the username, and the user concerned has the “UserName" *user* (the default one in this implementation). Note that the password is not visible since it is encrypted with the algorithm specified by the field “EncryptionAlgorithm". When using another security policy as ‘Aes128-Sha256-RsaOaep’, the whole message content is hidden: neither the requested service or the user information can be seen.

In summary, even if the information visible is limited to the username and potentially the OPC-UA address space (including the application value) when using the security policy ‘None’, it is important to fully understand the scope of this attack. Indeed, the attacker was considered here in the production cell, but the same attack could be carried out by anyone having access to the shop floor network, or even from the enterprise network in the case where the firewall is not correctly configured. The recommendation would be at least to not set up this security policy to ‘None’ even if it can slightly decrease the performance. Indeed, security implies an increase of the processing time, in particular for managing the encryption/decryption steps, and potentially increase the costs in terms of energy consumption and computing resources. It would be worthy to evaluate the overall cost of security. Is it not somehow the price to pay?

2.4.2 Message flooding & impact on an OPC-UA application

In the second attack, the attacker plans to disturb the smooth running of the application. To do so, he/she uses a TCP SYN flood attack that consists of sending as much as possible TCP connection requests on a specific port (here, the port 12686) so as to either congest the network or to cause a deny of service on the server.

Our objective is to show the impact of this attack on the OPC-UA application, and in particular at the client side. A packet sniffer (here *TCPdump*) is therefore running on the client in order to collect its communication with the server. *A posteriori* analysis has been led for computing the response inter-arrival time; i.e. the time between two OPC-UA responses, that should be in theory equals to the requesting period 10 ms (if the processing time is equal to 0). However, in practice, we observed an inter-arrival time between 14 ms and 40 ms with only one client and without any concurrent traffic (neither normal or abnormal traffic), meaning that the server can take between 4 ms and 30 ms to process the request (including the network delay).

Figure 2.3 shows the inter-arrival time (as defined previously) distribution over 10 experiments. The whisker diagram represents respectively the minimum value, the 10th percentile, the median, the 90th percentile and the maximum value of the series, that consists of 1000 requests (and 1000 responses) in total. Overall, it

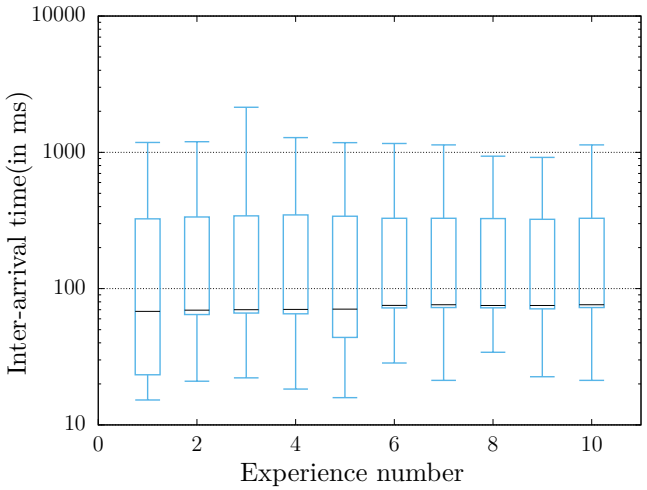


Figure 2.3: Inter-arrival time distribution over 10 experiments (y-axis in log scale)

can be observed that 80% of the values (i.e. between the 10th percentile and the 90th percentile) are between 70 ms and 120 ms (at least for 8 out of 10 experiments), and that the maximum is even worse (approximatively 1 s). The attack has therefore an important impact on the inter-arrival time. Those values would be unacceptable in case of critical applications (e.g. control of a process). In addition, this increasing time is also due to the relatively high number of TCP retransmissions (between 11% and 26.4%). It means that some frames have been lost (all the requests have still received a response, sometimes after 3 TCP retransmissions) due to the inability of the network (switch and/or server) to process them. As expected, this is the result of the competition on the output port of the switch to access the OPC-UA server and also the result of the increase of the processing time (since the server has much more frames to handle, and therefore a more important CPU load). However, a deny of service situation is not really reached since the attack is running from only one computer - *a distributed deny of service (DDoS) would have been more powerful from an attacker perspective, by using several computers at the same time for overloading the server.*

Looking now at a complete experiment over the time (Figure 2.4). Let us assume that the attack was launched after a while (and not at the beginning of the experiment) and for some reason stops before the end of the experiment. At the beginning (from 4 s to 22 s), normal inter-arrival times are observed. But suddenly, the inter-arrival times are increasing significantly from 22 s to 58 s of simulation,

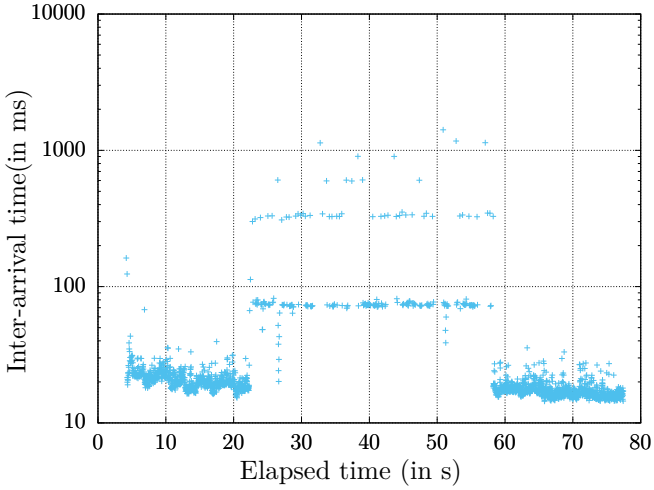


Figure 2.4: Inter-arrival time over time for one experiment (y-axis in log scale)

revealing an anomaly from an application perspective, even if it is nonetheless difficult to conclude that it is an attack (from the client side). Indeed, this effect can be due to a device/software misconfiguration or babbling idiot failure. This experiment also shows that if it is possible to stop quickly an attack, the OPC-UA server is able to quickly recover to respond to the client again with acceptable response times (in particular, without server crash).

The next section therefore discusses about the solutions and/or standards enabling to improve the security of an Industry 4.0 network in general (not only from an OPC-UA perspective, that already implements relevant mechanisms as explained previously) but also the next attack vectors.

2.5 On the way of using TSN for the Industry 4.0: new attack vectors & potential countermeasures

As explained in the introduction (cf. Section 2.1), the trend is to make the Operational and Information Technologies convergent in the Industry 4.0 environment. A first level of interoperability is achieved by using OPC-UA that acts as a middleware between the network layers (lower layers) and the application. A second level of interoperability could be used at the lower layers. In that sense, the Institute of Electrical and Electronics Engineers (IEEE) working groups, such as

the Audio Video Bridging (AVB) and more recently the Time-Sensitive Networking (TSN), are working on solving this challenge. Ethernet will therefore fulfil the common hardware requirements (for this interoperability). The TSN working group has already published several standards (and others are still projects) as a subset of the IEEE 802.1 standard. The focus is on the hard real-time and reliable communications. Overall, this technology relies on a Time Division Multiple Access (TDMA) mechanism, enabling to guarantee deterministic data transfers in specific and reserved timeslots. Other timeslots will be shared between different type of services (from network control traffic and alarms/events to best effort traffic). To manage those timeslots, the devices will rely on the time sharing according to the IEEE 802.1AS standard, that is a subset and superset of IEEE 1588 v2 standard (i.e. including a Precision Time Protocol (PTP) profile).

Time will therefore play a major role in this technology. As a matter of fact, it will become a new attack surface. Indeed, message flooding attacks (as shown in the previous section) could still be used, but would enable to overload "only" a timeslot - *that could be already enough for disturbing some applications* -. As this time synchronisation is based on the exchange of PTP messages (that do not implement security mechanisms), those messages could be used by a malicious user to falsify the time reference in the network. For instance, the attacker could use the same mechanism as in the Section 2.4.1, i.e. by using an ARP spoofing technique not only for eavesdropping the communications but also for modifying the (PTP) message itself. This can somehow be seen as a perspective for improving either the IEEE 802.1AS (PTP) protocol itself or the current network protection mechanisms (e.g. firewall). That is also a reason why in practice, the architecture still remains divided into different zones (as seen in the Figure 2.1) even if there is OT/IT convergence by the use of Ethernet (everywhere).

Let us note nonetheless that security mechanisms at the Layer 2 of the OSI model are being specified by the TSN working group. For instance, one can cite the project P802.1Qci (Per-Stream Filtering and Policing) that will enable to check whether a frame fits a reserved timeslot. Different strategies would be defined: i) limiting the bandwidth either by stream or by class of traffic or ii) blocking the traffic. The first one would enable to mitigate the impact of an abnormal traffic (message flooding/DoS attack or babbling idiot failure) whereas the second one would enable to completely stop it (and quickly to come back to a normal situation). A this time, it seems that the second one (blocking) would be preferred for standardisation to be sure to contain the abnormal situation.

In addition, 802.1AEcg (MACsec) could be used for maintaining data confidentiality between different authorised systems (interconnected to an Ethernet network). By using it, all the frames are authenticated and (optionally) encrypted. It enables to prevent attacks such as eavesdropping or message replay. Of course,

the overhead introduced in this mechanism need to be taken into account when defining the TSN timeslots. The analysis of the performance of using this mechanism in a TSN network can also be seen as an interesting perspective (even if it is not directly related to security).

2.6 Conclusion

Within the framework of overhauling the Industry 4.0 network, where the Operational Technology and Information Technology tend to be merged, OPC-UA is a promising solution to solve the important interoperability issue at the application level. However, when integrating such a protocol into the industry, it is important to be aware of the implemented security mechanisms (and their potential lacks). That is why this paper presented an identification - *based on the standard specifications and related documents* - of the threats and countermeasures/mitigations and an overview of the impact of those threats on an OPC-UA application (by using a real implementation). Overall, OPC-UA as middleware, implements satisfactory security mechanisms, but can not fill the lack of 'security-by-design' of legacy technologies (e.g. fieldbuses consisting of distributed I/O without OPC-UA capabilities). Consequently, it is also important to tackle the security issues at the field level. This paper would not have been complete without talking about the new standardisation progress for the future Industry 4.0, and its new potential attack vectors and countermeasures. The integration of TSN have therefore been discussed, and showed that future implementation would need to be careful to new attacks, and in particular against the time management. Finally, it will be important to keep a close eye on the development and standardisation of new mechanisms (e.g. IEEE 802.1AEcg, IEEE 802.1Qci) that could be interesting for security purposes.

Conclusion

This chapter/study has a focus on the protocol itself and from a security point of view, the protocol fulfills its role as an interoperable and secure interconnection framework. Note that from a performance point of view the protocol is also suitable for an industrial application [BRT19]. However, in order to have a completely secure data collection infrastructure, the previous study as well as the risk assessment study highlighted the need of additional external security mechanisms in the IT infrastructure that are more practical such as the the addition of new Access lists, the modification of the firewall rules, the inclusion of Intrusion Detection Systems (IDS).
Another important consideration is that since this study has been conducted, we are witnessing more and more PLCs natively implementing OPC-UA, however companies must ensure that the security mechanisms are implemented in the controllers. Indeed, in our study we used an open OPC-UA version that follows the standard specification. But due to the complexity of the standard, some controller manufacturers could simplify some parts, including the security mechanisms.
Finally, this study reassured Cebi regarding the adoption of OPC-UA as an interconnection framework. Nowadays, OPC-UA is used at Cebi for publishing data to various applications.

Part II

Flexible Process Control

Foreword

In the previous part, the focus was on the security of the interconnection framework that enables the IT/OT convergence and consequently the ability to access data from anywhere in the infrastructure. This provides the ability to explore the use of data-driven approaches such as Artificial Intelligence (AI)-based approaches for different applications. There are numerous use cases in manufacturing for such approaches, ranging from stock optimisation to predictive maintenance. However, most of these use cases are applicable at a factory or business level. However, with the advent of the Industry 4.0, we are witnessing a paradigm change in the way the controls of the processes are designed and the emergence of more and more open and software-oriented PLCs such as OpenPLC [ABdSR14] or SOEM[a] which is an open-source EtherCAT master. We are witnessing also some AI applications leading to 'Self-X' properties, such as self-configuration or self-repair. From a process optimisation perspective, the following question arises:

At the production cell level, could AI (potentially enabled by the aforementioned new PLCs) help improve the *flexibility* of manufacturing processes?

As mentioned in the Introduction (1), *flexibility* is one of the key features of the Industry 4.0. As a matter of fact, *flexible* production lines and process controls allow to enable productivity gain by enabling evolutions to the production line (e.g., change the products that can be produced on a line when the line is down). It also allows the processes to adapt to a changing environment/setting without having to manually modify the controllers' programs. Reaching this *flexibility* property requires to change the way the processes are controlled and consequently the way the PLCs are programmed. To do so, drawing on the paradigm change in the way the processes are controlled, we propose to explore to what extent the integration of Machine Learning (ML) in the control architecture could help improve flexibility. Our intuition is that it is necessary to step away from the static Programmable Logic Controller (PLC) programs and replace them by ML methods, which could help the production lines to adapt themselves to an evolving environment and even allow the lines to self-configure.

Such a study requires a data collection infrastructure and a safe testing environment in order to conduct the experiments. However, because at the time of the study the data collection infrastructure was not ready at Cebi and because we could not provoke downtime inducing revenue losses for the partner, we conducted experiments on scale model of a factory. To ensure the realistic and representative nature of the experiments, we chose actual

industrial PLCs to control the factory as well as new advanced PLCs so as to study their capabilities to integrate ML at the edge for various applications. Additionally, we focus on a representative Time Series Classification (TSC) problem one could encounter in the industry and highlight the implications and limitations of using such a method from a technical point of view.
This chapter presents our study published in a peer-reviewed journal - *MDPI Sensors* - in 2020 and entitled *"A Case Driven Study of the Use of Time Series Classification for Flexibility in Industry 4.0"* [PRLT20b].

[a]https://github.com/OpenEtherCATsociety/SOEM

3

A Case Driven Study of the Use of Time Series Classification for Flexibility in Industry 4.0

Contents

3.1 Introduction

Over the last few years, the manufacturing sector has been facing a new industrial revolution. This fourth industrial revolution — Industry 4.0 — originates from Germany, and was initially a proposal for developing a new concept of German economic policy in 2011 [RMK16]. Enabled by different technologies and concepts, such as the IoT, data-mining, and ML algorithms, new decision-making processes, CPS, Operational and Information Technologies convergence (OT/IT) [PRLT19], the main features of the Industry 4.0 are the following [PTB+15, LFK+14, Jaz14, VHH16, BMR+20]:

- Customisation/Individualisation of the products: the production needs to adapt to the customer's requirements that tend to be increasingly precise and individual. This also allows for the development of innovative business models.
- Flexibility: the production chain needs to automatically adapt itself to the evolving requirements (e.g., the range of products it should produce or the speed of the conveyer), to enable productivity gains.
- Product traceability: one needs to be able to identify all of the processes through which raw material, parts, and products have been modified/realised/assembled.
- Optimisation of the production process: thanks to the IoT devices and the information gathered and analysed by machine learning algorithms, it is possible to have a precise overview of the production process and, thus, optimise resources and, for example, predict forthcoming maintenance actions.

Meeting these expectations requires that European manufacturing companies rely on more advanced Information and Communication Technologies (ICT) bases and integrate IoT (or IIoT) at a larger scale, as it is considered to be a key enabler of Industry 4.0 [RMK16]. IoT — which describes the network of physical objects embedding sensing capabilities and software components in order to exchange data over the internet — is, in fact, an overarching term that applies to all of the sectors that can be digitised using physical sensors. IoT thus encompasses many of the current and future technologies and concepts [LL15], such as smart cities where different devices (e.g., sensors, cameras, etc.) are installed in a city in order to monitor the vehicular traffic, measure and report the environmental pollution, provide information on the weather, guide the car drivers into parking lots with the most space available, and so on [AHL+16]. Another important area of IoT applications is smart grids. Smart grids are electrical power grids that are smarter than traditional power grids thanks to IoT technologies. These smart grids self-heal, distribute electricity on demand, are monitored and controlled in real-time, and they rely on several small power producers, which makes the grids more efficient than traditional grids [SFL17, BTJ20, YY10]. IoT is also used in

healthcare, for example, to monitor patients' health conditions [UG15], by collecting different information, such as electrocardiogram (ECG), blood pressure, etc. It allows for reducing the risk of errors and, thus, improving the efficiency and lowering the costs of healthcare services [TAM16]. Industry 4.0/Smart manufacturing falls in the list of sectors that are covered by IoT, and, since it heavily relies on IoT technologies as enablers. IoT technologies allow for new applications and improvements in the manufacturing world, such as IoT-based cloud manufacturing, cyber-physical manufacturing, energy efficiency management, supply chain, and logistics traceability improvements, as reported in [YKBT19].

Unfortunately, most European industries — especially companies having decades of existence — do not have up-to-date ICT and automation infrastructures. Indeed, current manufacturing systems still rely on the Computer-Integrated Manufacturing (CIM) model, where the production systems run as independent systems (i.e., without any communication with the upper levels). The brain of such production systems is the PLC, which is an industrial computer monitoring its inputs (sensors) and outputs (actuators) in a cyclic manner. It makes decisions by itself based on the internal program logic. Therefore, this program is static and rigid in a sense that the programmed routines cannot evolve over the time unless a human modifies the PLC program. It makes difficult or almost impossible, for instance, to widen the range of products manufactured on a same line without having to stop it and load another program each time that we change the range of products. It would be the same difficulty for all major functional or environmental changes, e.g., a variation of the production speed or a move of the production system to a place where the lighting environment is different would require a long manual configuration phase. So far, this is achieved by blue-collar workers (i.e., the operators) that monitor, operate, and maintain, in operational condition, the production lines.

Working together with an automotive parts manufacturer — Cebi Luxembourg S.A. — through an R&D partnership, our main objective is to improve the overall efficiency of the production systems while shortening the "Measure Decision Act" loop. Thanks to a lot of sensor data (also known as Time Series (TS)) being collected, ML algorithms can ingest them for monitoring the factory's overall performance in real time, as well as understand and predict technical issues. However, in order to further improve the flexibility, we are convinced that it requires moving away from the aforementioned old-fashioned and rigid automation to a ML-based automation, i.e., where the control itself is based on the decisions that were taken by ML algorithms. We propose exploring whether and to what extent such a ML-based automation approach can replace a manual and static setting of a production line configuration (i.e., where all of the parameters have been manually set for the limited range of products).

Therefore, the contributions are threefold: (i) we concisely review time series classification methods and their applications in the industry, (ii) we apply a selected time series classification method from the literature to a real use-case developed on a scale model of a factory (FischerTechnik factory simulation) relying on a real ICT infrastructure (detailed in the next section), as it is not possible to stop a production line for deploying such a scenario without beforehand demonstrating the feasibility, and (iii) we discussed the implications and limitations of using such an approach from a technical point of view in the industry, to serve as knowledge base on how to evolve towards such an approach in real-settings.

The rest of this paper is organised as follows: Section 3.2 explains the problem, Section 3.3 gives a review on time series classification methods and their implementation in the industry, Section 3.4 presents our case study and experimental setup. Experiments have been conducted in order to improve the flexibility of production lines and the results are presented in Section 3.5. Section 3.6 discusses our results and methods. Additionally, finally, Section 3.7 concludes the paper and presents ideas for the future.

3.2 Problem Statement

Our main goal is to improve the flexibility of the production lines, i.e., the ability of the line to automatically adapt itself to evolving requirements (a larger variety of parts to produce, for example). However, European manufacturing companies usually rely on the CIM model. Consequently, each production system has its own logic that is implemented in a PLC. PLC programs are developed in a rigid manner, and they are not meant to evolve in order to adapt to a changing environment. Any modification to the code requires human intervention. This old-fashioned and rigid automation model is a hurdle to the improvement of the production lines' flexibility. As an example, Figure 3.1 is an excerpt from a PLC program of a real production line (the whole program contains over 6000 instructions, each line is an instruction). Modifying this program requires having a perfect understanding of the process and the program, since the sequence of instructions is important and it should not be randomly modified (it could impact the process). Each modification implies a large testing process on the production line.

Consequently, to tackle the limitations of static programs, we propose exploring to what extent these static PLC programs could be replaced by a state-of-the-art ML-based method. To do so, we first need a safe development environment, in which we are able to experiment and test without damaging anything or modifying the processes in the factory. Indeed, the industrial partner cannot afford the downtime that it would require to test this solution; he needs to understand the benefits of using such a method and evaluating the Return On Investment (ROI), and finally, we can also propose a more innovative architecture using new

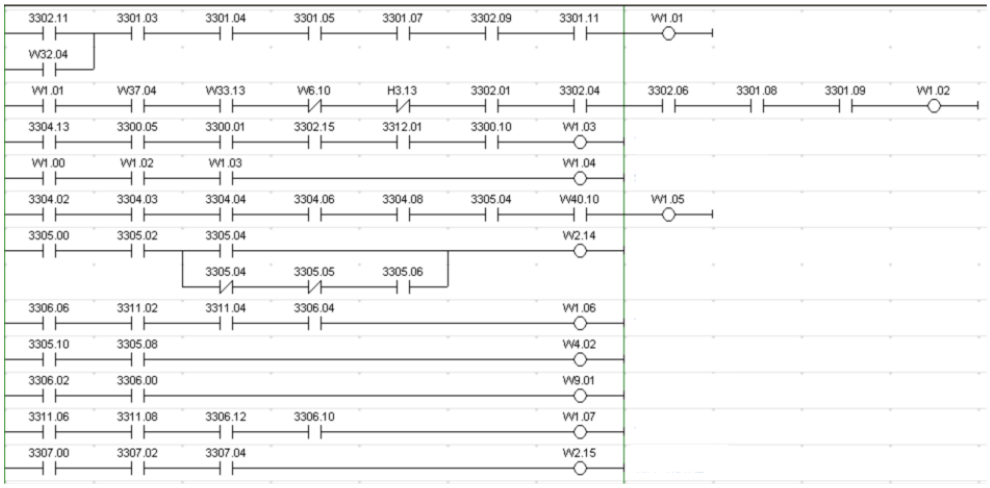


Figure 3.1: Ladder program from a traditional controller.

generation PLCs in a development environment. Another key factor in the choice of our development environment is its use in scientific papers. Finally, we need to find a use case that complies with several properties: (i) it should be representative, i.e., common in the industry, such as an anomaly detection process, a sorting process, or more generally a classification process, (ii) it should be minimalist and canonical, i.e., based on one variable, if it works with one variable and a small amount of data (which is a limit case for ML), it will most likely work with more variables.

The *FischerTechnik* 24 V factory simulation (FischerTechnik Factory simulation website) has been chosen. This platform has been used in the literature for different objectives, such as CPS security research hackathons [FAB+18], in order to develop a new low-cost, but powerful, controller [VBLS20], or to develop a multi-agent control system [BCMC08], and the automation architecture (controllers, sensors, and actuators) is representative of a real production line, since it relies on real industrial controllers to control the different stations that compose the platform (our architecture is detailed in Section 3.4). The four stations are: (i) the automated warehouse where parts are stored, (ii) the multi-processing station (simulating an oven and a saw), (iii) the vacuum gripper robot that is used to move the parts between the warehouse and the multi-processing station, and (iv) the sorting line with the colour sensor that is used to sort the parts according to their colour.

On this platform, we focused on the fourth station, namely the sorting line. This choice is justified by the fact this station is the one that best fits our case by using a colour sensor returning numeric values, on which a decision is taken under

temporal constraints. Thus, this case is more interesting, while the other stations rely on position sensors or limit switches, which return binary values. As a matter of fact, it is composed of three sensors: an input sensor, a colour sensor, and an output sensor, and three actuators to store the part in the right stock according to its colour. Based on those inputs/outputs, the program logic implemented in the controller is as follows: the value of the colour sensor is collected every 10 ms from the moment the part is detected by the input sensor, until the moment the part is detected by the output sensor. By doing so, we are able to create time series, such as the ones depicted in Figure 3.2, where the sizes range from 350 to 500 observations per TS due to the fact that parts are going from a conveyer belt to another, and this could take more or less time. The blue TS is from a blue part, the red TS is from a red part, and the grey TS is from a white part (the platform comes with blue, white, and red parts).

This is a traditional TSC case. The next section gives a background on TSC methods and their application in industry.

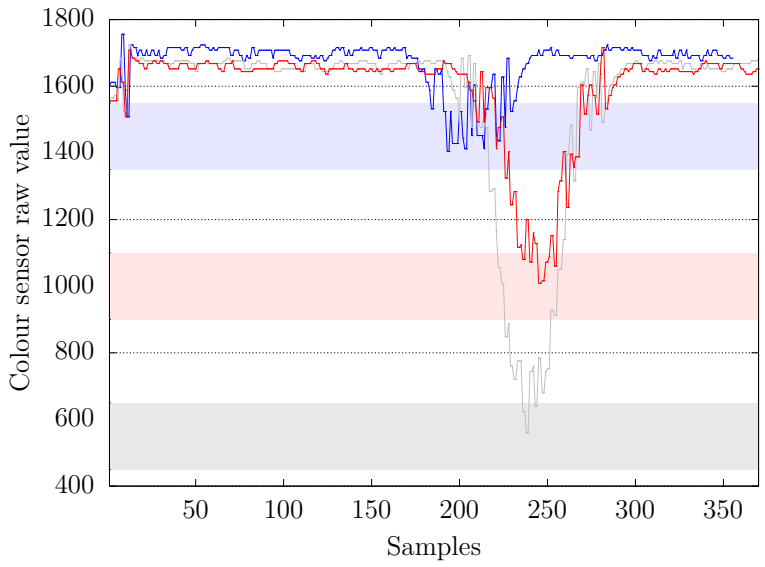


Figure 3.2: Time series of the three colours & thresholds.

3.3 Background & Related Work

3.3.1 Time Series Classification

An univariate time series (TS) is an ordered sequence of values that were taken by the same variable, recorded at fixed sampling rate. Time series are increas-

ingly generated and analysed in different application domains, such as finance (e.g., stock markets' fluctuations), health monitoring applications (e.g., electrocardiograms), meteorology (e.g., weather forecasting), and industry (e.g., sensor values) [DTS+08, FJ14, Li18, SL17]. The multiplication of application areas is driven by, among other things, the increasing use of sensors and IoT devices in several domains, as denoted in the previous subsection (e.g., 61.8% of the UCR datasets are sensor data or simulated data) and the fact that increasing researchers and practitioners are using time series to model non-temporal observations [Li18] (e.g., 38.2% of the UCR datasets are data from images).

The objective of TSC is to assign a class label to a TS that is based on a set of predefined classes. Basically, a set of labelled TS is learned by an algorithm, and the algorithm applies what it learned to label each non-labelled TS of a testing set. The TS classifiers can be divided into three categories:

- Whole series similarity-based methods [BLB+17]: these methods measure similarity or dissimilarity (distance) between two entire TS and usually use a 1-NN classifier in order to evaluate the distance. The most common methods are 1-NN Euclidian distance (ED) and 1-NN Dynamic Time Warping (DTW [BC94]), [EA12]. Several other algorithms have also been proposed, such as Weighted DTW (WDTW), Time warp edit (TWE), or Complexity invariant distance (CID), as denoted in the review that was written by A. Bagnall et al. [BLB+17].
- Feature-based/Symbolic representation-based methods: these methods usually compute features from the TS and then compare the features that were computed from the training TS to the features computed from the test TS. These methods can be subdivided into two types: shapelets-based methods and Bag-Of-Patterns (BOP)-based methods. Shapelets, as introduced by L. Ye & E. Keogh [YK09], are subsequences of TS that are the most representative of a class. Subsequently, J. Lines et al. proposed a Shapelet Transform (ST) [LDHB12] in order to improve the classification accuracy by extracting the the best k shapelets for each class and avoid overfitting. The result of this transform can be used as an input of traditional classifiers. Several other shapelets-based methods exist, such as Fast Shapelets (FS) or Learning Shapelets (LS). On the other hand, Bag-Of-Patterns (BOP) is a histogram-based similarity measure, which is similar to Bag-Of-Words for text data [LKL12]. BOP uses a sliding window to extract subsequences of the TS of a fixed length, and then each subsequence is converted into a Symbolic Aggregate approXimation (SAX) [LKWL07] string.
  The most accurate classifier for combining with BOP-SAX is 1-NN using Euclidian distance [SM13]. P. Senin & S. Malinchik proposed the Symbolic Aggregate approXimation-Vector Space Model (SAX-VSM), which uses a

different feature weighting system from BOP-SAX [SM13]. P. Schafer introduced the Bag-of-SFA-Symbols (BOSS) model [Sch15], which uses the Symbolic Fourier Approximation (SFA) [SH12] instead of SAX. P. Schafer & U. Leser then presented the Word ExtrAction for the time Series cLassification (WEASEL) [SL17] method, which is a BOP-based method that uses a specific method in order to derive features, so as to be fast enough to satisfy the requirements of sensor-driven application and still give accurate classification results. WEASEL will be further explained in the next section. Note that Bag-Of-Patterns-based methods have a linear complexity.

- Ensemble classifiers: these methods use a set of classifiers and aggregate the results while using different methods to return a classification result. Ensemble methods are the most accurate methods, yet they are computationally very expensive, since each classifier inside the ensemble has to be run. J. Lines & A. Bagnall proposed the Elastic Ensemble [LB15] (EE), which is an ensemble including elastic measures, such as DTW, variants of DTW, and ED. The same authors presented the Collective of Transformation-Based Ensemble (COTE) [BLHB15], which is an ensemble of 35 classifiers. The authors then proposed HIVE-COTE in 2016 [LTB16], which is an improvement of COTE while using a hierarchical probabilistic voting structure.

The next subsection presents the review on the application of such TSC methods in the industry.

3.3.2 Time Series Classification in Industry

With the *Industry 4.0* paradigm comes the convergence of the Internet Technologies and Operational Technologies, and concepts, such as Industrial Internet of Things (IIoT) (which is the IoT applied in an industrial environment), cloud manufacturing, and so on. These concepts and the ever-growing number of deployed sensors [SL17] are pushing industries into the big data age [TQLK18]. To exploit the full potential of this increasing quantity of data, practitioners are more and more interested in deploying ML-based methods which will help them to analyse the data and extract useful information related to the processes, the equipment, the production and so forth. However, this is not limited to manufacturing and it can be extended to other fields, such as smart-grid [WLPB18, OEV+15], smart-cities [AHL+16], transportation [MRR+17], health, sport, etc.

There are plenty of papers in the literature whose authors are working on implementing time series classification in industry on different use cases. Table 3.1 references those papers and summarises the approach used, whether it is a traditional ML-based or Deep Learning (DL)-based approach, the kind of data they use, and the area of application. We focused on papers on manufacturing and application domains that could have to deal with the same problems, and detailed what they do in the field (e.g., anomaly detection, health management, and so on).

To wrap-up this part of the related work, the trend seems to be the following: the papers using univariate TS are more focused on classification, monitoring, and anomaly detection, generally descriptive analysis and use ML-based methods. More complex problems, including prediction, often use ML-based approaches and multivariate TS or fusion data. Finally, prognostics and health management problems tend to use deep learning-based methods and multivariate TS. Deep learning methods are becoming more common for time series classification, but these methods are used as a black box. In our case, provided that we want to preserve the expert knowledge and, when considering our application, we will focus our interest on ML-based methods for univariate time series classification and, more precisely, the WEASEL method.

3.3.3 WEASEL

In order to use time series classification methods in our case study, we needed a method that is able to deal with TS with variable lengths, which are sometimes noisy, because they are issued by an inaccurate sensor (in our case, but it could also be due to the dust or luminosity in a real industrial environment); additionally, the classification time should be fast enough to still be able to sort out the parts in the right stock.

WEASEL has been proposed for addressing the challenges that are associated with sensor-driven applications [SL17]; namely, this method is robust to noise, able to scale to the number and the length of TS, able to classify TS with different sizes and different offsets, and does not need to know *a priori* the structure of the characteristic pattern. According to this description, this method *a priori* fulfills our requirements for such a case study. That is the primary reason why we select and apply this method in this paper.

Table 3.1: Implementation of time series classification (TSC) on industrial cases.

Paper	Approach		Method	TS Type		Domain	Application
	Trad.	DL		Uni.	Multi.		
[MLE+17]		X	LSTM autoencoder + Deep FeedForward Neural Net.		X	Process planning	Quality defects detection
[BCMM18]		X	Multi-Layer Perceptron		X	Transportation	Driver identification
[FKM17]	X		Decision trees, SVM, kNN, Random Forests		X	Smart buildings	Energy quality classification
[ÓVCB+18]	X		ED, FR, DTW, MDTW + similarity forests	X		Telecom.	Churn detection
[BCMM17]		X	Multi-Layer Perceptron		X	Gaming industry	Bot detection
[SVGCPM17]	X		wDTW + 1-NN	X		Sport	Pitcher's performance analysis
[LWT+18]		X	Deep Neural Net.		X	Manufacturing	Lithium-ion cell selection
[AMY+18]	X		MASS + kNN		X	Agriculture	Chicken welfare assessment
[MSF19]	X		Their method (Clustering + similarity tree)	X		Manufacturing	Self-learning + classification
[HY17]		X	Deep Belief Net. for feature leaning + LSTM		X	Health	Sleep patterns classification
[PC19]	X		Random Forests		X	Electrical industry	Short-term voltage stability assessment
[ZHR+18]	X		Principal Component Analysis + Bayesian Neural Net.		X	Climatology	Atmospheric new particle formation prediction
[ATS]	X		Statistical feature extraction + Bayesian classifiers		X	Manufacturing	Robot execution failures classification
[MATR19]		X	Convolutional Neural Net.		X	Manufacturing	Tool wear prediction
[NAR+20]		X	Convolutional Neural Net.		X	Transportation/safety	Drivers' emotions/behaviour classification
[GWJ17]		X	Long-term recurrent convolutional LSTM		X		Pattern recognition and forecasting
[BPP+19]		X	Convolutional Neural Net.		X	Cosmology	Supernovae classification
[RK19]	X	X	Random Forests (baseline), LSTM-RNN, MS ResNet, TempCNN		X	Satellite imagery	Vegetation modeling and crop type identification
[WLPB18]	X		Their method (clustering)	X	X	Manufacturing/Power grid	Anomaly detection
[LLZS18]	X		Wavelet decomposition + SVM		X	Manufacturing	Machining condition monitoring
[ZWYG18]		X	LSTM		X	Manufacturing/Prognostics	Remaining useful life prediction
[CXZ+20]		X	Convolution recurrent Neural Net. (CNN + LSTM)		X	Manufacturing/Prognostics	Bearings health indicator construction
[MMZ16]	X		kNN		X	Manufacturing/Prognostics	Remaining useful life prediction
[LZSW19]	X		Random Forests, AdaBoost, CART, RR, SVR, RVFL		X	Additive manufacturing	Surface roughness prediction

In addition, WEASEL is a great trade-off between good accuracy and fast training and prediction time. WEASEL was the second most accurate method after the COTE ensemble, but, when it comes to training time and prediction time, the few classifiers that are faster than WEASEL are less accurate, as demonstrated by the authors by comparing their method against the best core classifiers on all the UCR datasets [SL17]. This feature (prediction time) is important in our case, and it constitutes a second reason to select the WEASEL method.

WEASEL uses normalised sliding windows of different sizes in order to decompose the TS, and then it approximates each window while using the Fourier transform. Subsequently, the statistical ANOVA F-test is applied to keep the Fourier coefficients that allow for differentiating the TS from different classes. Those kept coefficients are then discretised into words while using information gain binning. A Bag-Of-Pattern is built from these words and encodes unigrams, bigrams, and the windows. The Chi-Squared test is applied to the Bag-Of-Pattern to reduce the feature space and only keep the most discriminative words. Finally, the classification is done by logistic regression.

To the best of our knowledge, only a few works have been done while using WEASEL. M. Middlehurst et al. [MVB19] evaluated WEASEL to find out whether it could be a good replacement for BOSS in the HIVE-COTE ensemble. The authors compared the WEASEL, BOSS, and BOSS variants; it turns out that WEASEL is the most accurate method, yet it is the slowest to build on average. W. Sousa Lima et al. applied BOSS, BOSS-Vector Space (BOSS-VS), SAX-VSM, and WEASEL on human activity data sets from three databases and compared them regarding the accuracy, the processing time, and the memory consumption [SLdSBMQPS18]. Even if WEASEL has never been implemented in an industrial context, it seems to be promising for our use-case, and it is selected for discussing an innovative and flexible automation based on traditional ML. Note that the method is not compared with other ML/TSC methods in this paper, but only with the most effective and efficient one at the time of the study. For a comparative study, we redirect to the one that has been conducted in [SL17], which demonstrates that WEASEL outperforms a baseline of ML algorithm for TS classification. In this paper, building on the best existing algorithm that is found in the literature, the goal is to study to what extent it can be applied successfully, the limits in using it, and the adaptations (here preprocessing) that are required to optimise it to a typical Industry 4.0 problem. We believe that this would be a natural and intuitive choice to take the best algorithm in any real-world setting.

The originality of this work is in the fact that we design a more flexible automation architecture, integrating a TSC method from the literature, which we evaluate by conducting live tests. By conducting live tests, we are able to demonstrate the applicability of such a method in an industrial production process, in

order to show that this method allows a more flexible production and to highlight the limits and adaptations that are required to implement such a method.

3.4 On the Use of ML in the Automation Architecture

European manufacturing companies usually rely on the CIM model, where each production system has it own logic implemented in a PLC, as mentioned in Section 3.2. Such programs are often developed in a cyclic manner; i.e., by successively reading sensors' inputs, processing them, and finally updating outputs (actuators). There is no (or very few) intelligence, which makes them rigid. To make them a bit more flexible, one way would be to develop functions that would be triggered by API calls to create the current logic. This would also enable introducing external ML-based programs while keeping industrial PLCs in the automation architecture. To go even further, those PLCs could be replaced by industrial (micro) computers having more resources for using more advanced/intelligent functionalities. In that sense, we designed the architecture of our *FischerTechnik* factory simulation, so as to demonstrate this possible evolution.

It consists of three controllers/PLCs that possess their own sets of inputs/sensors and outputs/actuators. Each PLC implements its own program logic by using the programming languages that are related to the hardware. For instance, a Function Block Diagram and/or Sequential Flow Chart is used on the Crouzet em4 (Crouzet website) for controlling the warehouse. Note that this PLC implements several functions that are triggered by a central controller while using the Modbus/TCP protocol. For controlling the robot, we used a Controllino Maxi PLC (Controllino website) (based on arduino) and, for the multi-processing station and the sorting line, we used two Wago EtherCAT slaves with an EtherCAT master that was implemented on a Raspberry PI3. Those latter use C-based programs, and they have been chosen to show the evolution in terms of automation architecture.

First, we defined a manual and static setting of a production line configuration for the sorting line. This represents the type of static programs implemented in old-fashioned controllers. The next section will provide more details.

Let us remind that our objective is to advance from a static setting of the program (such as shown in Algorithm 1, where the variable *ts* is a vector of n integer values v_j and th_i are integer thresholds (i = 1,2,3,4,5,6)). Accordingly, instead, the idea is to have a dedicated agent that is capable of predicting/classifying a time series according to a trained model (that can evolve easily over the time). In that architecture, the controller can implement the program logic that is defined in Algorithm 2, where *ts* is a vector of n integer values v_j and where line 3 is an API call to this dedicated ML-based agent. Note that this agent can run directly

on the same device or another (in a distributed architecture).

Algorithm 1: Static program.

```
1: ts ← [v1,v2,...,v_n]
2: minTs ← min(ts)
3: if th_1 < minTs < th_2 then
4:     colour ← blue
5: else if th_3 < minTs < th_4 then
6:     colour ← white
7: else if th_5 < minTs < th_6 then
8:     colour ← red
9: end if
10: sort(colour)
```

Algorithm 2: Flexible program using ML prediction.

```
1: ts ← [v1,v2,...,v_n]
2: colour ← getPrediction(ts)
3: sort(colour)
```

In our testbed, this agent uses WEASEL and its "official" Java implementation v.3. It was implemented on a HTTP server that was hosted by a MacBook Pro with a 2.8 GHz Intel Core i7 processor. When the values are collected and the TS is generated, it is sent to the server by an HTTP client (as part of the main program logic) on the Raspberry PI3 (depicted as the prediction request in Figure 3.3), then the server executes WEASEL on the TS and sends its class prediction (depicted as prediction response in Figure 3.3) and, based on the class, the controller sends the command to the actuators to push the part in the right stock (depicted as sorting order in Figure 3.3). Thereby, the colour verification and the sorting are done "on-the-fly".

Before being able to use WEASEL@running for classifying new parts (and, therefore, sorting them out), the use of ML requires collecting time series as

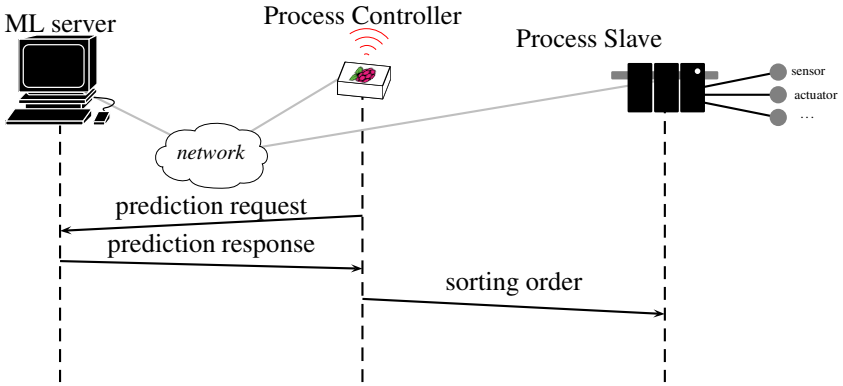


Figure 3.3: Scenario architecture.

datasets for training and evaluating our models (cf. Figure 3.4). Although the data collection, the training, and the evaluation processes can be automated, a human intervention is still needed to label our data (i.e., assigning the right class/-colour at each time series). In the same way, and according to the accuracy of our model, misclassifications of the test TS can be re-labelled for re-training the model and improving it.

This pipeline has been evaluated on our testbed. The next section provides the results and discussions.

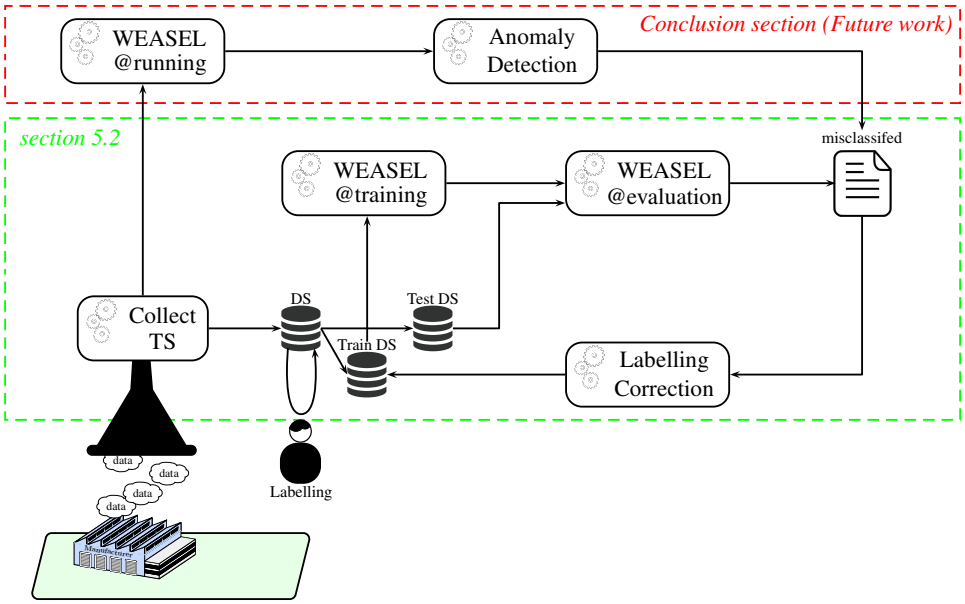


Figure 3.4: semi-automated (with Machine Learning) integration process.

3.5 Models Evaluation

Figure 3.5 presents the methodology defined for conducting our experiments and evaluations. Additionally, all of our experiments are conducted by logging each TS and the corresponding class that is predicted by WEASEL on the controller, in order to save the testing data sets for offline replays, for example. In addition, parts have always been following a predetermined sequence, so as to know the ground truth and to be able to evaluate the results.

We evaluate the results while using the classification accuracy indicator, which is computed using Equation (3.1).

$$Accuracy = \frac{number\ of\ correctly\ classified\ instances}{total\ number\ of\ instances} \tag{3.1}$$

Note that we do not use the other metrics, such as precision, recall, and F1-measure, since we have balanced datasets, which means that the classes inside of the datasets contain the same proportion of samples (e.g., in the three-colour test datasets, 33.3% of the parts are blue, another 33.3% of the parts are red and the last 33.3% are white, and the costs of misclassifications are the same whether they are false positives (i.e., for the blue class, an actual white or red part classified as a blue part) or false negatives (i.e., an actual blue part classified as a red or white part). In this specific case, the classification accuracy metric itself is meaningful and sufficient, while it may not be the case when the datasets are imbalanced and, when, for example, false negatives are more important than false positives, such as in the study of the propagation of a virus.

The study has been driven following three research questions (RQ), namely, RQ1: can WEASEL (off-the-shelf) be as accurate as manually set thresholds on verifying and sorting the three original colours? RQ2: Is WEASEL able to handle evolutions? RQ3: Is one single generic model accurate enough or is a model for each set of colours mandatory?

3.5.1 Preparation

Before evaluating both our static program and our models, the first imperative is to collect data. For each six colours we had (three originals — *blue, white, red* — and the three others we 3D printed — *orange, green, purple* —), 100 time series (TS) have been collected; this makes a total of 600 TS where the sizes range from 350 to 500 samples each. The labelling is done manually with classes between one and six. The first 100 TS are blue; therefore, the class assigned is one, the second 100 TS are white, so the class assigned is two, so on and so forth. This step corresponds to the first activity, denoted A1 in Figure 3.5.

Following the different steps of the program logic integration (Fetch TS, Analyse them, Build program, & Integrate it), thresholds have been defined. The defini-

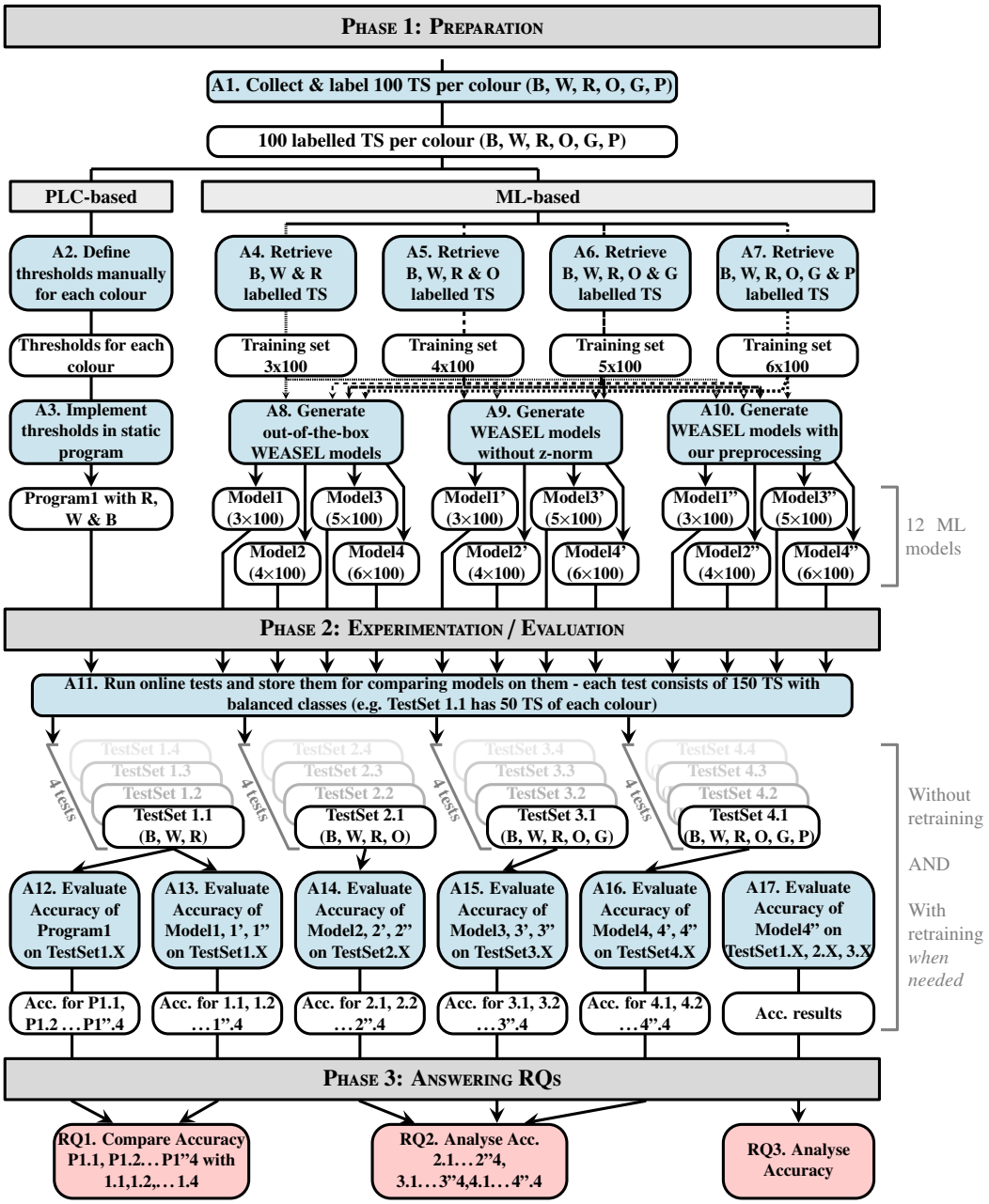


Figure 3.5: Methodology for models preparation, creation, and evaluation: the blue boxes correspond to the activities, the white ones to the input/output, the grey ones to the different phases, and the pink ones to the analyses for answering the research questions.

tion of the threshold has been done based on the minimal value range, as depicted in Figure 3.2, except the 100 TS for each colour were used to determine the range. With these defined thresholds, the program logic that is shown in Algorithm 1 has been built. This type of logic is what is implemented in old static controllers. These steps correspond to activities 2 and 3, denoted A2 and A3 in Figure 3.5.

The next important step is to create the ML models. In order to represent the evolution of the production in terms of variety of product, we chose to create models trained on three colours, four colours, five colours, and six colours. In order to train such models, training datasets are created by fetching the labelled TS (A4 to A7 in Figure 3.5). Therefore, a total of four training datasets was obtained. The first one contains 100 TS per colour and three colours, so it is basically composed of 300 TS. The second one is the same as the 300 TS training set, in which we added the 100 TS of the fourthxs colour, so it is composed of 400 TS. The third one is the same as the 400 TS training set, in which we added the 100 TS of the fifth colour. Finally, the fourth one is the same as the 500 TS training set, in which we added the 100 TS of the sixth colour. For each set of colours, three types of models are generated, namely out-of-the box models (with default parameters) denoted *ModelX*, models without preprocessing denoted *ModelX'* and models with our preprocessing method denoted *ModelX"*. Assuming that WEASEL (out of the box) could be not effective enough in our case where the major differences between the TS reside in their amplitude (as shown in Figure 3.6) due to its intrinsic z-normalisation method, we investigate it with this normalisation disabled. As the TS amplitudes are relatively close between classes, we propose a preprocessing method, so as to separate them. This makes a total of 12 models that will be evaluated and compared:

- **Four out-of-the-box models** (cf. activity A8 in Figure 3.5): in order to observe how effective the out-of-the-box configuration is, four models were trained. As mentioned before, WEASEL natively uses the z-normalisation as a preprocessing method. This method ensures that the output vectors (TS) have a mean around 0 and standard deviation close to 1. That means that the output TS on which the models are trained will have the same range of amplitudes. *Model1* is a model that is trained with default parameters on the three-colour 300 TS data set. *Model1* took 17.0 s to train. *Model2* is trained on the four-colour 400 TS data set and took 64.9 s to train. Following this pattern, *Model3* is trained on the five-colour 500 TS data set and took 92.2 s to train, and *Model4* is trained on the six-colour 600 TS data set and took 112.6 s to train, still with default parameters.
- **Four no-preprocessing models** (cf. A9 in Figure 3.5): In order to verify whether the z-normalisation method is effective enough in a case where the major differences between the TS reside in their amplitude, four models

were trained the same way, except the default z-normalisation applied in WEASEL has been disabled. This means that the models are trained on raw data. By doing so, the amplitude of the TS will become an important feature. Accordingly, *Model1'* is trained (without z-normalisation) on the same dataset as model *Model1* (3×100 TS, three colours), *Model2'* on the same dataset as *Model2* (4×100 TS, four colours), *Model3'* on the same as *Model3* (5×100 TS, five colours) and *Model4'* on the same as *Model4* (6×100 TS, 6 colours). The elapsed times in the training process for *Model1'*, *Model2'*, *Model3'*, and *Model4'* are, respectively, 15.7 s, 24.5 s, 71.7 s and 96.0 s.

- **Four models with our preprocessing** (cf. activity A10 in Figure 3.5): For the purpose of concluding on the importance of a suitable preprocessing method, four models were trained with our preprocessing method. Let us first present the preprocessing method that we propose. This method is for ensuring the same reference for the TS inside of the same class. Because the means of the TS inside of the same class are the same, but they are different between each class, we chose the mean as the reference for the TS. Our preprocessing is as follows: (i) for each TS, we compute the mean value of the whole series, and subtract this mean value from each original element of the time series; (ii) for each modified element, we compute its absolute value, and replace the element by this absolute value (this reduces the brutal changes in the TS caused e.g., by the inaccuracy of the sensor measurement or any environmental change). Using this method, we not only ensure that TS from the same class have the same reference (their means), but we preserve the amplitude difference in the pattern and we also deliberately shift TS from different classes (as depicted in Figure 3.7 between y = 0 and y = 200, as opposed to y = 1600 in Figure 3.6), and the reference is also becoming a characteristic. We implemented this method in WEASEL's code and trained the following models. *Model1"* is trained on the same training set as *Model1* and *Model1'* (3×100 TS, three colours), *Model2"* on the same training set as *Model2* and *Model2'* (4×100 TS, four colours), *Model3"* on the same training set as *Model3'* and *Model3* (5×100 TS, five colours), and *Model4"* on the same training set as *Model4'* and *Model4* (6×100 TS, six colours). The elapsed times in the training process for *Model1"*, *Model2"*, *Model3"*, and *Model4"* are, respectively, 17.6 s, 31.0 s, 157.6 s, and 243.4 s.

3.5.2 Experimentation-Evaluation

For evaluating the models and static program, testing data sets are needed. However, as already stated, the classification has to be done online, so we conducted an online evaluation of some of the models, and stored the testing data sets in order to compare the accuracies of different models on the same test sets.

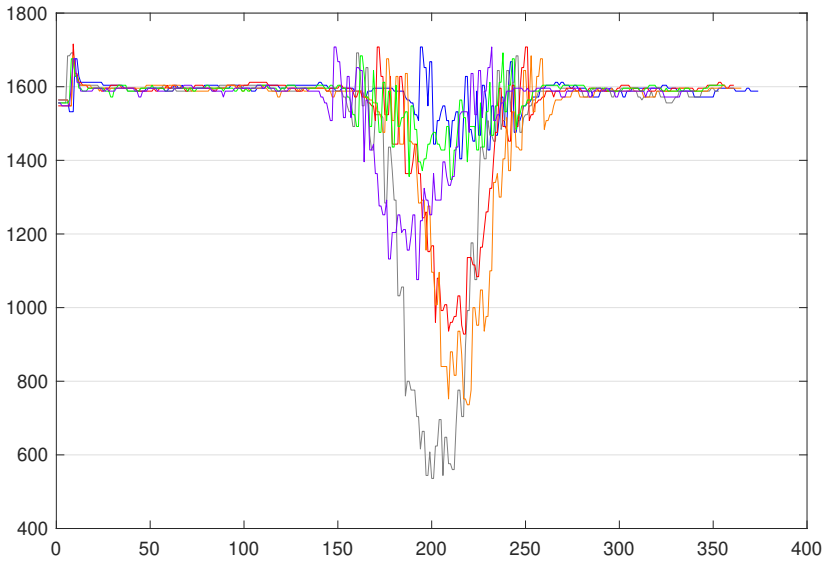


Figure 3.6: 6 colours TS with no preprocessing.

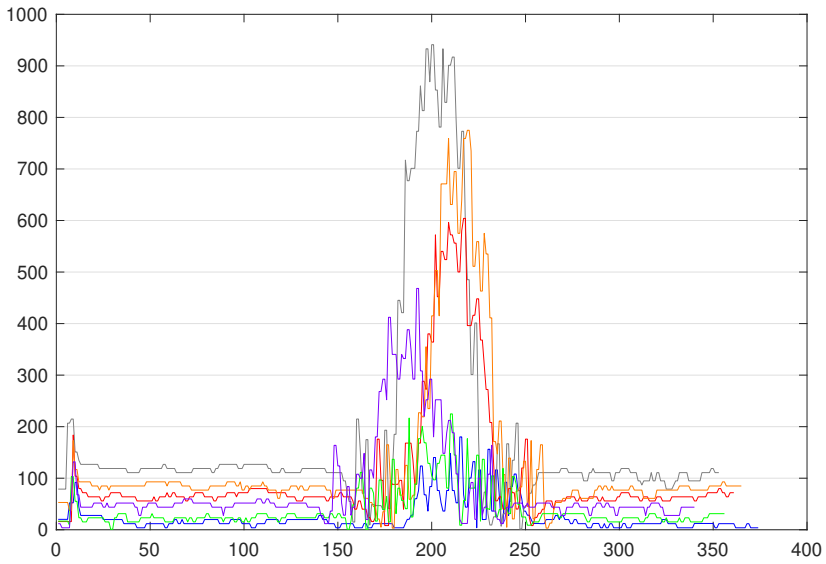


Figure 3.7: 6 colours TS after preprocessing.

Overall, we gathered four series of four test sets, such as that depicted in Figure 3.5 (cf. A11). Each series of four test sets is composed by TS of different

colours, i.e., the first series (*TestSet1.1* to *TestSet1.4*) is composed of TS of three colours, the second series (*TestSet2.1* to *TestSet2.4*) contains TS of four colours, the third series (*TestSet3.1* to *TestSet3.4*) contains five colours, and the fourth series (*TestSet4.1* to *TestSet4.4*) contains six colours. Each test set, regardless of the number of colours, is made of 150 TS, with balanced classes.

The first actual experiments (to answer RQ1) are the evaluation of the static program, *Model1*, *Model1'*, and *Model1"* (cf. A12 and A13 in Figure 3.5) on *TestSet1.1* to *TestSet1.4*. The accuracies are detailed in Table 3.2. By looking at those results, we can see that the static program is efficient and the native z-normalisation is not the best normalisation for a case where TS from different classes have the same patterns, but different amplitudes, since the z-normalisation normalises the TS, such that they have the same amplitude. *Model1* still gives satisfying results in this case.

Table 3.2: Classification accuracies of the static program and the models on three colours.

Test Set	**1.1**	**1.2**	**1.3**	**1.4**
Static Program	100%	100%	100%	100%
Model1	99.3%	100%	99.3%	96.7%
Model1'	100%	100%	100%	100%
Model1"	100%	100%	100%	100%

The next experiment is the evaluation of *Model2*, *Model2'*, and *Model2"* (cf. A14 in Figure 3.5) on *TestSet2.1* to *TestSet2.4* (4 colours). The accuracies are presented in Table 3.3. Here, we can clearly see that the z-normalisation (*Model2*) is not optimal and our preprocessing (*Model2"*) gives satisfying results.

Table 3.3: Classification accuracies of the models on four colours.

Test Set	**2.1**	**2.2**	**2.3**	**2.4**
Model2	93.3%	86.7%	86.7%	81.3%
Model2'	100%	100%	100%	100%
Model2"	98.7%	96.7%	97.3%	99.3%

The next experiment is the evaluation of *Model3*, *Model3'*, and *Model3"* (cf. A15 in Figure 3.5) on *TestSet3.1* to *TestSet3.4* (5 colours). The accuracies are presented in Table 3.4. With these results, we can see that neither the z-normalisation (*Model3*) nor no normalisation (*Model3'*) are optimal. This is due to the fact that

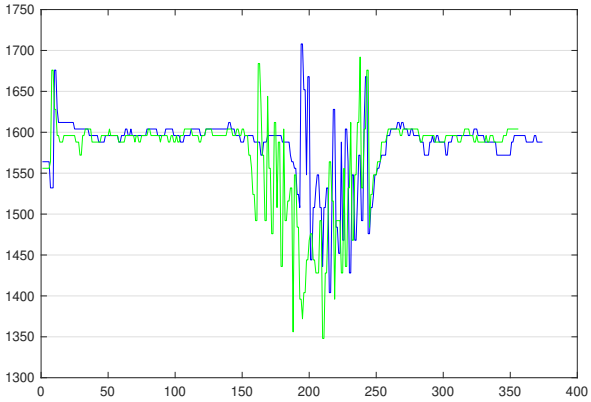


Figure 3.8: Blue and green time series.

some TS of different classes have the same amplitude, e.g., blue and green TS, as shown in Figure 3.8. However, our preprocessing method, since it adds an offset between TS of different classes, allows for WEASEL to classify TS with a better accuracy.

Table 3.4: Classification accuracies of the models on five colours.

Test Set	**3.1**	**3.2**	**3.3**	**3.4**
Model3	64.7%	60.7%	66.7%	80.7%
Model3'	46%	40%	40%	44%
Model3"	98%	98.7%	98%	99.3%

However, updating the training set with misclassified TS from previous tests and retraining the model allows to increase the accuracy. With *Model3'*, we took the misclassified TS from *TestSet3.1*, we corrected the label, and then added them to the training set, so the set contains 581 TS. Subsequently, we replayed the test on *TestSet3.2* with the updated *Model3'*, and we reiterated the retraining process (the full process is shown in Figure 3.4 — green box). *Model 3'* is now trained on 587 TS, and the accuracies are shown in Table 3.5.

The next activity (A16 in Figure 3.5) is the evaluation *Model4*, *Model4'*, and *Model4"* on *TestSet4.1* to *TestSet4.4* (6 colours). The accuracies are presented in Table 3.6. In this scenario, we can see that the models with or without z-normalisation (i.e., *Model 4 and 4')* give very low accuracies and a large deviation between the test sets can be observed. By contrast, *Model 4"* accuracies are close to satisfactory results, and there is room for improvement with retraining.

Table 3.5: Classification accuracies of Model3' after the retraining process.

Test Set	**3.1**	**3.2**	**3.3**	**3.4**
Model3'	46%	40%	40%	44%
Model3' updated once (581 TS)	/	96%	*nn*	*nn*
Model3' updated twice (587 TS)	/	/	97.3%	100%

/ means test not done since TS used to retrain the model. *nn* means not needed.

Table 3.6: Classification accuracies of the models on six colours.

Test Set	**4.1**	**4.2**	**4.3**	**4.4**
Model4	52.7%	22.6%	35.3%	74%
Model4'	24.7%	31.3%	20%	33.3%
Model4"	94.7%	79.3%	79.3%	100%

Model4" has therefore been retrained using exactly the same process as previously. We relabelled the misclassified TS from the test on *TestSet4.1*, added them to the training set which now contains 608 TS and retrained *Model4"*. Then we ran a test on *TestSet4.2* with the updated *Model4"*, and took the misclassified TS from this test, relabelled them and updated the training set a second time. *Model4"* has been retrained on the updated training set containing 621 TS, and we conducted tests on *TestSet4.3* and *TestSet4.4*. The accuracies are shown in Table 3.7. We can see that retraining the model increases the accuracy on the test with *TestSet4.3*, but slightly decreases it on the test with *TestSet4.4*.

Table 3.7: The classification accuracies of Model4" after the retraining process.

Test Set	**4.1**	**4.2**	**4.3**	**4.4**
Model4'	94.7%	79.3%	79.3%	100%
Model4" updated once (608 TS)	/	91.3%	*nn*	*nn*
Model4" updated twice (621 TS)	/	/	95.3%	98.7%

/ means test not done since TS used to retrain the model. *nn* means not needed.

The final experiment (A17 in Figure 3.5) consists of the evaluation of *Model4"* (since it is the one with the best results) on all of the test sets, except the ones with six colours, since it has already been done. This experiment aims at verifying whether it is possible to train only one "complete" model accurate enough to correctly classify parts, even if some sort of parts are missing, e.g., the model is

trained to classify products x, y, and z, but z is out of stock for some time, is it possible to keep the xyz model or is a xy model needed? Detailed accuracies are given in Table 3.8. All of the results are satisfactory.

Table 3.8: Classification accuracy of Model4" on three, four, and five colours.

Test Set	**1.1**	**1.2**	**1.3**	**1.4**	**2.1**	**2.2**	**2.3**	**2.4**	**3.1**	**3.2**	**3.3**	**3.4**
Model4"	100%	100%	100%	100%	98.7%	96.7%	98%	99.3%	97.3%	98%	97.3%	100%

3.5.3 Answering RQs

Based on the results that are presented in the previous subsection, RQs can now be answered.

First, **RQ1** is the following: can WEASEL (off-the-shelf) be as accurate as manually set thresholds on verifying and sorting the three original colours? The off-the-shelf WEASEL models we tried are not as accurate as the static program on classifying the original three colours; however, they still give satisfying results. On the other hand, WEASEL models without z-normalisation or with our preprocessing method are as accurate as the static program.

Second, **RQ2** is the following: is WEASEL able to handle evolutions? WEASEL is able to handle evolutions in terms of number of different colours/parts. However, out-of-the-box models may not be the most accurate when adding more parts, and the preprocessing method used has a strong impact on the results. Finding the most suitable preprocessing method (as we did) is needed in order to obtain the best accuracy possible. The results can still be improved by updating the training set and retraining the model.

Finally, **RQ3** is the following: is one single generic model accurate enough or is a model for each set of colours mandatory? According to these results, it is possible to use a single model that can accurately classify parts, even if some of the colours are included in the training set, but not in the testing set. In this case, that means that we do not need to generate a model per range of parts, which means that there is no reconfiguration to be done if the line needs to produce/sort/verify a sub part of the entire range that it can produce/sort/verify. Note that this may only be true for unidimensional feature spaces, high dimensional feature spaces cases may behave differently.

3.6 Discussion

In the previous sections, we demonstrated that a specific TSC method, namely WEASEL, which is integrated into an automation architecture, is able to enhance the flexibility of the production lines. Indeed, we showed that: (i) the TSC method

can be as accurate as a static program, so the effectiveness of the classification is not decreased; (ii) the TSC method is able to handle changes in terms of number of different colours/parts with some adaptations to the preprocessing method used before training the model; and, (iii) a single model trained on all of the parts can only classify a sub-part of the total variety of products. Of course, some kind of expertise is required in order to successfully apply such a method, especially with the preprocessing method, and there are limits and threats to validity, which are discussed in this section.

First, we chose to train the models with the maximum number of TS that we had for each colour. It might not be the most efficient method, since, for example, for the three colours model (blue, white, red), WEASEL without normalisation only needs 10 TS of each colour to reach 100% average accuracy on the four experiments. However, for the four and five colours experiments, the model trained on 100 TS of each colour is the most accurate.

Additionally, we could observe the colour sensor limitations, when considering that the range of values that it can return is quite narrow (0–2000 mV), it may not be possible to add more colours, because the TS corresponding to these colours will overlap (see when there is only one TS of each colour for the six colours in Figure 3.6). Note that, nonetheless, it also occurs in our experiments, in particular with blue and green colours. That is the reason why (i) it is really difficult to manage in a static program (where thresholds need to be defined) and (ii) the ML-based program was experimented.

Furthermore, we observed that a single model generated with all the data (six colours) is accurate, even when classifying only three, four, or five colours. This allows efficiency gains, since the same model can be used at all time, there is no need to change the model if some parts are not produced/sorted/verified for some time. However, as the classification is done on-the-fly, it may be important to look at the impact of the model size on the prediction time. If the model size increases, the prediction time to the point where the classification result is given so late that the controller cannot send the instruction to the actuator to push the part before the part is in the front of the right stock, then it is better to use a smaller model if the variety of parts is reduced (even temporarily).

Note that, even though an experimental evaluation of the prediction time has not been detailed in this paper, online tests allowed for us to witness that all of the parts have been sorted on time, which means that the prediction time is suitable.

Finally, regarding the preprocessing methods we investigated, we could observe that the z-normalisation is not suitable in our case, since the amplitudes of the TS need to be retained for the TSC method to correctly classify them. We also observed that training the models on raw data is not suitable either. When applying our proposed preprocessing method, we witness that the models training

time increases exponentially with the number of TS in the training set, while it increases linearly with the other two methods. However, our preprocessing method allows for accurately classifying the parts. Note that the training time is not as important as the prediction time in this case study, which was not impacted by the preprocessing method.

Additionally, we are aware that the degradations we could observe under these conditions while using the WEASEL method could have been different with another TSC method. However, the adaptations that we propose in terms of data preprocessing before training models could be suitable for other methods.

Overall, using ML-based methods instead of static programs presents numerous advantages and plays an important role in the improvement of the flexibility of production lines. Some of these advantages are the following:

- ML-based methods allow for distinguishing patterns in time series, even when a human and, thus, a static program cannot distinguish them
- They allow for changing the variety of parts to be produced without having to change the program and, consequently, without having to stop the production line (provided that the model has been trained on data for all parts, obviously)
- They allow for processing a large amount of data

However, we could notice that these methods are not flawless. Indeed, they also have some drawbacks:

- The methods require expertise to tune them; they are not accurate enough out of the box
- Decisions can be made too late if the method is not fast enough, since they are made on a remote agent (request and response transmission time, and remote code execution time have to be considered)
- Some of the methods require several days to build a model (e.g., DTW-based methods on a lot of data)

3.7 Conclusions

In order to improve *flexibility* in today's static industries and help the manufacturers to optimise the Overall Equipment Effectiveness, we proposed using a ML-based method instead of static programs and demonstrated the feasibility of applying such a method in the industrial context based on a scale model of a factory. This method gives promising results and allows for increasing the range of coloured parts to be sorted on the same sorting line when it would be impossible to manually set thresholds to distinguish two colours. Indeed, we extended the use of WEASEL in order to apply it on time series with almost the same patterns and only the amplitude differing, but this requires some expertise and comes at the cost of a reduced robustness against environmental changes. In order to tackle this

robustness reduction problem, we proposed modifying the preprocessing method used in WEASEL. Finally, we showed that, in our case, the production line could be even more flexible and uses only one model whether it is to produce/verify/sort the complete variety of parts or only a sub part.

The way that we implemented the WEASEL method might not be the best, even when disabling the z-normalisation method, because it has a negative impact on the robustness, and we had to propose another preprocessing method in order to mitigate the impact. Additionally, we faced a few issues on such a simple canonical case; issues might be larger and more complex when applying such a method on real manufacturing processes.

For future research, we will consider the automation of the retraining process, as our use case is based on the colour verification and sorting of coloured parts, we know, *a priori*, the sequence of parts that have to be sorted. We will also take a closer look at prediction times, especially when we use a complete model for classifying a sub part of the whole variety of parts. Moreover, we will explore whether the ML-based method is also able to handle changes in the production speed. Finally, we will add some security mechanisms, so as to build a trusted complete framework, as described in Figure 3.4.

Conclusion

This chapter focuses on the integration of traditional ML methods in the process control architecture based on a representative TSC problem on a scale model of a factory. This allows for future improvements in flexible process controls, with different methods, at least on similar use cases. Additionally, this demonstrates that using new generations of PLCs with open communication capabilities allows for the use of ML at the edge, by using a ML server as an Application Programming Interface (API) for example, as long as the prediction delays are suitable for the application. It is important to note that in such a context, since the control is done remotely, it is important to evaluate the robustness of the ML methods to perturbations that can happen in the network between the data production and the ML server. This is the focus of another work-package of the Research/Development and Innovation (RDI) project with Cebi and conducted in another PhD thesis [BRLT21].
This study paved the way for Cebi to start using ML in the control architecture, not for flexibility purposes yet, but for quality control.

Part III

Blockchain-based traceability

Foreword

Precise *product traceability* is another key feature of the Industry 4.0 and one of the biggest challenge in the manufacturing industry. Indeed, keeping track and trace of the different processes and operations - *whether they are quality controls or settings and configurations of the machines/tools* - during the product life cycle requires having a secure data collection infrastructure, as discussed in Chapter 2 and is crucial to prevent high-cost recalls and provide the customers with precise information and circumstances when recalls happen. As an example, Takata - *a Japanese airbag maker* - had to recall tens of millions of defective airbags inflators that were used by nineteen car manufacturers in 2017[a]. As they were not able to identify precisely the incriminated products, there are still, to this day, defective inflators in some cars on the road, and still incidents caused by this. The costs of this recall led to the company bankruptcy. Another example to illustrate the expenses caused by some of these recalls, it costed $4.1 billion to General Motors in 2014 in order to cover repair costs, victims compensations and other expenses related to recalls[b]. This shows the importance of relying on a trustworthy traceability system to identify the responsibilities.
However, an effective traceability emphasizes some challenges:

1. there needs to be a product-data model, which aims at defining a unique product identification method as well as the product and process related information to integrate. This requires a perfect understanding of the products (their composition, assembly, etc.) and the processes.

2. a trustworthy and reliable data collection and storage architecture, which aims at ensuring that the data is not manipulated nor modified from the production of the data to its storage.

In this part of the dissertation, we focus on the second challenge, more precisely on investigating a trustworthy data storage architecture. The data collection part has been partly explored through the first chapter of this dissertation (Chapter 2) in which we assessed the security mechanisms proposed by OPC-UA and in which we concluded that OPC-UA associated to some other protection mechanisms such as Intrusion Detection Systems (IDS) and firewalls play a major role for a trustworthy data collection infrastructure. However, as mentioned in the Introduction (1) and highlighted by [MSS18], building a trustworthy traceability system is challenging and expensive and the return on investment is difficult to measure. Additionally, in order to avoid data tampering and single points of failure, it is required to step away

from traditional centralised traceability systems [XLL+19]. One alternative is emerging in both the industrial world and the academic world for this purpose: Distributed Ledger Technology (DLT) and blockchain technologies.
As a matter of fact, these technologies allow to **store data in an immutable way**, which is an important asset, especially when the system is shared between various participants and some of them are not trusted. To illustrate the interest around blockchain and DLT, for example, the car manufacturers Renault, BMW, Ford and General Motors launched in 2018 a research group on blockchain and mobility services named Mobi. This group is composed of around thirty industrial actors, start-ups and governmental agencies[c]. Another example could be the company Hyundai working on a project of using blockchain to track used vehicles history[d]. Finally, the number of consulting firms writing technical reports on 'Enterprise Blockchain' and also reporting statistics on the different platforms adoption in the enterprises show the interest of industrial actors in blockchain technologies. For example, Cognizant Technology Solutions wrote a survey based on 281 respondents from the manufacturing industry entitled *"Blockchain in Manufacturing: Enhancing Trust, Cutting Costs and Lubricating Processes across the Value Chain"*[e] in which they (amongst other interesting information) list the platforms tested by the different companies.
Considering these examples, we can imagine in a near future these big manufacturing companies/groups asking their suppliers to also be familiar with blockchain technologies and share their data with them through their blockchain. For this reason, it is of utmost importance for these suppliers, such as our partner Cebi - *supplier for several car manufacturers* - to conduct a technological watch on blockchain and DLT. However, there are a lot of different blockchain platforms, implementing different features, designed for various applications. The choice of a platform is therefore difficult. Then, there are a lot of settings to choose prior to the implementation of the platform and the impact of such settings is not always straightforward to apprehend. In addition, the infrastructure required to test and size the blockchain architecture can become very costly (in terms of computational resources, human and financial resources, and so on) so the design phase might be a problem for numerous companies willing to deploy a blockchain platform, in particular when platforms should be completely implemented and running prior to any experiment.
This part investigates the use of blockchain for traceability purposes and is composed of three chapters. In Chapter 4, we identify and compare five of the major permissioned blockchain frameworks used in the industry, based on

the literature. This chapter presents our study published in a peer-reviewed journal - *ICT Express* - in 2021 and entitled *"Permissioned blockchain frameworks in the industry: A comparison"* [PRL21].
Then, in Chapter 5, we propose a methodology for selecting an optimal blockchain configuration and we instantiate this methodology on a private Ethereum platform implemented on our partner's servers. This chapter presents a study that is not yet submitted nor published.
Finally, in Chapter 6, we propose BlockPerf, a simulator/emulator framework designed to implement and to compare different blockchain platforms and evaluate it against BlockSim (a major simulator in the literature that BlockPerf actually extends) and against a real bitcoin blockchain implementation. This chapter presents our study published in a peer-reviewed journal - *IEE Access* - in 2021 and entitled *"BlockPerf: A Hybrid Blockchain Emulator/Simulator Framework"* [PGK+21].

[a] https://www.livemint.com/Companies/nfxY3ng1P0VULYeJ1qH5lK/Takata-bankruptcy-poses-a-risk-to-auto-industrys-biggest-re.html

[b] https://money.cnn.com/2015/02/04/news/companies/gm-earnings-recall-costs/index.html

[c] https://www.usine-digitale.fr/article/renault-bmw-gm-des-constructeurs-ouvrent-un-centre-de-recherche-sur-la-blockchain.N688519

[d] https://www.cryptopolitan.com/hyundai-to-track-used-car-history-through-blockchain-project-with-blocko/

[e] https://www.slideshare.net/cognizant/blockchain-in-manufacturing-enhancing-trust-cutting-costs-and-lubricating-processes-across-the-value-chain

4

Permissioned blockchain frameworks in the industry: A comparison

Contents

4.1 Introduction

Originally introduced with Bitcoin by Satoshi Nakamoto in 2008 [Nak08], blockchain is a distributed ledger technology, where participants maintain a replica of the shared ledger. The ledger updates and the replicas' synchronisation are maintained through a consensus protocol executed by the participants. The ledger contains cryptographically signed transactions which are put together in blocks. Using such cryptographic techniques makes the ledger immutable, which is one of its main characteristic.

Even though blockchain was first invented for building a public and open trustless network without any central authority, it is evolving towards permissioned and private platforms for enterprises. Private blockchains are similar to public blockchains, they also are immutable, nodes share the same ledger, but the access to the network is permissioned. This means that permission and role for each node have to be granted.

In this paper, the features of the major private blockchain frameworks are presented. Hyperledger, Ethereum, Quorum, MultiChain and R3 Corda are considered as the main technologies [for]. We compare them in terms of community activities, performance, scalability, privacy and adoption based on a literature review.

The paper is organised as follows: section 4.2 describes the main features of the selected frameworks. The comparison analysis is presented in section 4.3. Section 4 discusses our key findings. Section 4.5 concludes the paper.

Table 4.1: Major permissioned frameworks' features

	Industry Focus	Consensus	Smart Contracts	Open Source	Support/ Governance	Cryptocurrency
Hyperledger Fabric	①	Voting-based (Solo) or pluggable consensus	✓	✓	Linux Foundation	✗
Ethereum	①	PoW or Clique PoA	✓	✓	Ethereum developers	✓
Quorum	②	Clique PoA or RAFT-based or Instanbul BFT	✓	✓	Ethereum developers & JPMorgan Chase	✗
MultiChain	①	Round Robin validation	✗(v1) ✓(v2)	✓ / Com.	Coin Sciences	✗
R3 Corda	②	Voting-based / RAFT	✓	✓	R3 Consortium	✗

① Cross-Industry ② Financial Industry / Com.: or Commercial

4.2 Major Features: Hyperledger Fabric, Ethereum Geth, Quorum, MultiChain and R3 Corda

Even though these 5 platforms share the common property of being permissioned platforms, they all have their own features detailed in the following. Also, all the main features are summarized in Table 4.1.

Hyperledger Fabric is one of the Hyperledger projects hosted by the Linux Foundation. It is a decentralised operating system for permissioned blockchains that can execute distributed applications (Dapps) written in general-purpose programming languages such as Go, Java or Node.js [ABB+18]. A Fabric network is composed by nodes whose identities are given by a membership service provider. These nodes can be: i) *Clients* that propose transactions to execute and broadcast them for ordering; ii) *Peers* that maintain the ledger and the state of the latter; or iii) *Ordering service nodes* that establish the order of all the transactions. Note that the latter do not participate in the execution nor the validation processes. Additionally, Fabric uses smart contracts, called *chaincodes*, for implementing the application logic. The platform natively implements Solo, a voting-based consensus protocol consisting of the *endorsing nodes* (a subset of peers) executing the transactions and validating them in compliance with the *endorsement policy*. However, other consensus protocols such as Practical Byzantine Fault Tolerance (PBFT), Raft or Kafka (to be able to use several orderers) can be plugged in. Fabric has no underlying cryptocurrency.

Ethereum, launched in 2015, is an open source, public, blockchain-based, distributed platform for developing decentralised applications [eth]. Originally, Ethereum is a public permissionless blockchain-based platform implementing a Proof-of-Work (PoW) based consensus protocol called Ethash. Ethereum is also used as a private platform (configurable feature). As PoW is not secure enough and requires a lot of computational power, some of the Ethereum testnets are currently implementing the new Clique Proof-of-Authority (PoA) consensus protocol. All blocks are then sealed by approved signers, which is computationally lighter than Etash's mining process. The list of signers is dynamic in Clique: signers can propose to add or remove another signer and the proposal that gets the majority of votes is applied. Additionally, Ethereum supports smarts contracts written in the Solidity object-oriented language for running Dapps. Ethereum also introduces the Ethereum Virtual Machine (EVM) enables any node to run any program regardless of the programming language. Ethereum has a native cryptocurrency called Ether.

Quorum has been developed by J.P. Morgan for financial use-cases, but can used for any type of industry. Quorum is a permissioned blockchain based on the Ethereum blockchain [quo]. More precisely, it is a fork of go-ethereum. It brings several enhancements: i) *Privacy*: it is possible to create private contracts and

transactions whose payload is only visible to participants that are specified in a parameter of the transaction. Public transactions are still possible (i.e. visible by all the participants of the permissioned network, not by the public Ethereum blockchain); ii) *Alternative consensus protocols*: other protocols for consortium blockchains such as a Raft-based consensus protocol and Istanbul BFT are also available; iii) *Permissioning*: only known and authorised peers, defined in smart contracts, can join the network; and iv) *Higher performance*.

MultiChain, developed by Coin Sciences, is an open source platform which is a fork of the Bitcoin blockchain [mul]. However, unlike Bitcoin, MultiChain allow users to configure several parameters such as, e.g. the permissions to access the network, the privacy of the chain, the maximum block size, the mining incentive. The mining is done by a set of identified block validators. There is a single validator per block, working in a round-robin type of scheduling. In MultiChain 1.0, it is not possible to build complex logic on the blockchain because of the lack of support of smart contracts, but the new MultiChain 2.0, which is in Beta version at the time of writing, introduces *Smart Filters*, a functionality that enables custom rule coding for validating transactions. Finally, MultiChain supports a variety of programming languages such as Python, C#, PHP, Ruby or JavaScript.

R3 Corda is an open source permissioned platform developed by R3 [cor]. Corda follows the "Know Your Customer" principle, each node has to prove its identity to be authorised to join the network. The *Doorman* is the node in charge of validating the identities and distributing the certificates. The network is also composed by one or many *Notary* nodes, their role is to validate the uniqueness and the sequencing of the transactions without global broadcasting. Two types of consensus have to be reached in Corda: validity and uniqueness. Validity is checked by each signer before signing the transaction, and uniqueness is checked by the *Notaries*. Smart contracts written in JVM languages are supported, and Corda supports the development of decentralised applications (CorDapps) written in Kotlin.

4.3 Comparison

When selecting a private framework for a specific use-case, it is important to know several factors such as community activity, the technology adoption or even the intrinsic performance. In the following, the methodology for assessing each criteria and a comparison of the frameworks are presented. Note that our framework's performance analysis considers only their original/initial features (such as the intrinsic consensus) from a high-level perpective, i.e. without considering and reviewing every single detail that can be used or plugged-in. This is particularly true at consensus level where others consensus (e.g. BFT-like consensus, RAFT, ...) could be implemented.

4.3.1 Methodology

Community activity: To assess it, we mainly look at i) the Github repository of the frameworks in terms of number of contributors (denoted u) and commits (c) and ii) Twitter in terms of number of followers (f) and tweets (t) on the main account. We determine a final grade (gca) between 0 and 5 for each technology i such as defined by equation 1.

$$gca_i = \left\lceil \frac{\frac{c_i}{\max c} + \frac{u_i}{\max u} + \frac{f_i}{\max f} + \frac{t_i}{\max t}}{4} * 5 \right\rceil \tag{4.1}$$

Adoption: Can be measured by looking at industrial use-cases. We rely on i) the recent Forbes Blockchain 50 report [for], presenting the underlying technologies embraced by big companies of different sectors (such as Amazon, Facebook, Google, BMW, Daimler, ...) - *the number of groups embracing a framework is denoted n* - and ii) the scientific interest for each or these frameworks. This latter is measured by reporting the number of results to the search of the platform's name on Google Scholar (denoted h). Our searches are the following: ethereum, hyperledger fabric, quorum blockchain, R3 corda, "multichain" blockchain, and all of them do not count patents and citations. A final grade denoted ga is therefore determined by using equation 2.

$$ga_i = \left\lceil \frac{\frac{n_i}{\max n} + \frac{h_i}{\max h}}{2} * 5 \right\rceil \tag{4.2}$$

Privacy / Confidentiality: When looking at the platforms' privacy, we mainly concern ourselves with the transactions privacy and data exposure/confidentiality, which is more concerning since privacy is more related to participants, and we must know participants in permissioned frameworks. We determine a grade based on the privacy-preserving mechanisms implemented in the different frameworks (e.g. possibility to create private transactions/contracts, the way the transactions are going through the network and so forth). The mechanisms were found in the white papers or reference papers for each framework [ABB+18, eth, quo, mul, cor]. In private blockchains, data and transactions can be seen either by all the participants (0-grade) or only a part of them. The more restrictive the framework can be, the better the grade is. Also, 0 does not mean no privacy at all.

Scalability, Throughput & Latency: Those criteria levels are computed thanks to our literature review (16 papers). Figure 4.1 depicts our three-step methodology. Our corpus consists of 1303 papers collected in five main library databases: IEEE Xplore, ACM Digital Library, Springer, ScienceDirect, MDPI.

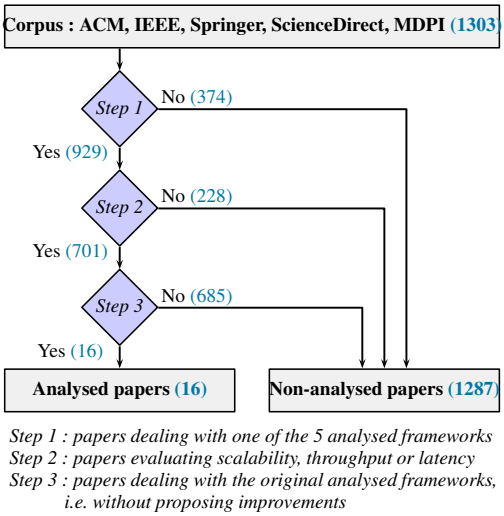


Figure 4.1: Methodology for gathering papers evaluating performance

These papers assess the performance of private/permissioned blockchain platforms. As a first step, we filtered only the papers dealing with one or more of the five frameworks under study. Then, only the papers evaluating our three metrics of interest are kept. Finally, titles and abstracts are analysed in order to exclude papers which i) claim results that are not obtained through an experiment, ii) implement a modified/improved version of the original framework, iii) evaluate on a domain-specific scenario (e.g. healthcare with non-comparable workloads/benchmarks) and iv) obtain results through analytical models (including simulated models). This final step significantly decreases the final number of papers to be analysed as listed in Table 4.2. Since no work compared all the frameworks in the same settings, we used the best results for each framework. Even if it constitutes a limitation of our comparison, it enables nonetheless to show to what extent the performance is reachable (already known and tested on a real set-up). To assign these grades, for each criterion, we look for the best framework and assign the best grade to it (5), then the other grades are assigned relatively to the best framework. For the latency metric, the 5 grade means the framework has the lowest latency.

4.3.2 Analysis

Figure 4.2 sums up the frameworks' behaviour with regard to the different criteria. Note that, as no paper dealt with the scalability criterion for Quorum and Multichain, the 0 grade is assigned. Looking at the average grade for each platform in order to compare them on one general grade (considering the same weight for each criterion). Fabric, Ethereum, Quorum, Multichain and Corda have respectively an average grade of 4.2, 3.3, 2.2, 2 and 2.2. Even if Fabric is, here,

Table 4.2: Analysed papers for the scalability, throughput and latency assessment

Framework	References
Hyperledger Fabric	[ZZL+18], [DWC+17], [SSAD19], [ABB+18], [HSGX19], [KPGR19], [HLD+18], [NQATN18], [PST17], [TNV18], [BSV+18], [SWTR18], [ARG+19]
Ethereum	[ZZL+18], [DWC+17], [TMOZ20], [HLD+18], [PST17], [RD17]
Quorum	*[BSKC18]**
MultiChain	[OCF+19]
R3 Corda	[HSGX19]

** Note that since no paper evaluated the performance of the Quorum framework, the following paper [BSKC18] was a posteriori added.*

showing better results, it is important to keep in mind that choosing a platform is mainly a trade-off. Indeed, none of these framework is flawless.

Our relative comparison is realistic, but we are aware of the following limitations: i) some experimental results in the literature are still missing regarding some criteria for several frameworks or conducted in different test environments; ii) the grades regarding these metrics might not reflect exactly the real behaviour of each platform (since only based on a limited number of scenarios experimented in the literature).

4.4 Discussions / Lessons learnt

Based on our literature review and the comparison analysis presented in the previous sections, we highlighted several key findings that can be interesting for both practitioners and researchers:

For industrial practitioners: The provided analysis can be used as a starting point for selecting 2 (or 3 at maximum) frameworks for testing in their own settings/environment without (re-)developing a complete analysis. This comparison give them important criteria to consider when implementing such blockchain frameworks. It is important to implement the latest version of the framework, since it is usually more secure, scalable and performant as pointed out by [NQATN18, HSGX19], that shows that Hyperledger Fabric v1.0 performs better than the v0.6. However, Fabric still suffers from scalability issues [DWC+17, HSGX19]. The performance of a DApp will also depend on the type of transactions (read and/or write) as demonstrated by [KPGR19, BSV+18] for Fabric. This study also highlights that practitioners have to be careful when configuring the queue size in such a way that it limits the increase of the latency when there are too many simultaneous transactions. Additionally, practitioners willing to implement a Fab-

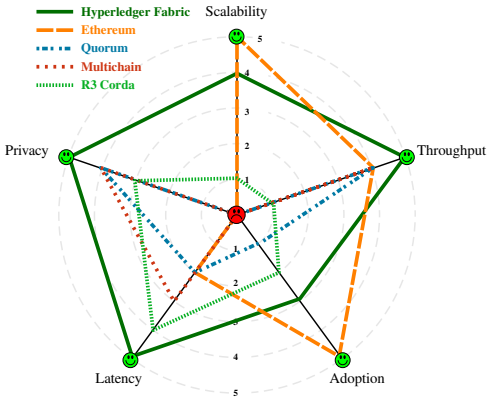


Figure 4.2: Overall analysis

ric platform need to be aware that the orderer's settings have an impact on the throughput such as demonstrated in [BSV+18], and the throughput is also affected by the choice of the database [TNV18, RD17]. On the other hand, when implementing an Ethereum platform, practitioners also need to consider the overhead in terms of memory and disk usage it implies [TMOZ20, DWC+17, RD17], and their vulnerability to attacks forking the blockchain, as denoted in [DWC+17].

For researchers: Our literature review highlights that: i) no work compares the major permissioned blockchain frameworks in a same environment, that could be very interesting for analysing deeper where improvements could be done; ii) no work presents a methodology for sizing the transactions queue, which has an impact on the latency as shown in [KPGR19], iii) there is still room for improvements regarding the consensus protocols used in the private platforms, since they often are the bottlenecks [HSGX19]; iv) there are plenty of optimisations that could be done/found for enhancing the performance of private platforms as shown and experimented in [TNV18, SSAD19] and v) there are still experimental studies to be conducted regarding privacy and scalability, particularly in Ethereum, Quorum and Multichain.

4.5 Conclusion

Private blockchains are quickly becoming a concern for both the academic and the enterprise worlds. And they are more and more used in the enterprises [for]. However, there are a lot of frameworks, and all of them are slightly different in terms of consensus protocols or underlying currency. We described the main features of five of the major platforms - Hyperledger Fabric, Quorum, Ethereum (geth), MultiChain and R3 Corda - so as to gather and highlight the differences between them in one paper. We analysed the frameworks with regard to several criteria using results obtained from experimental research papers and proposed

a comparison based on grades assigned to the platforms for each criterion. We could notice that most papers use old versions of the frameworks and some of the comparisons are debatable, whether it is because the hardware is different or because the features of the platforms are such that it cannot be compared. For future work, we will implement the latest versions of these platforms and experimentally compare them using different benchmark tools.

5

A methodology for selecting an optimal blockchain configuration

5.1 Introduction

In the previous chapter, we identified and compared five of the most commonly implemented permissioned blockchain platforms in the industry. The study highlighted the performance (among other characteristics) of the different platforms and allowed us to conclude on several points:

- There is no single best platform regarding our criteria
- There is no comparison of all these platforms in the same environment
- The settings/parameters of the platform impact the performance

Before implementing any platform and conducting a comparison study, or any performance evaluation of a single framework it is required to understand the impact of the different settings and to find an optimal configuration to reach the best performance for a specific application.

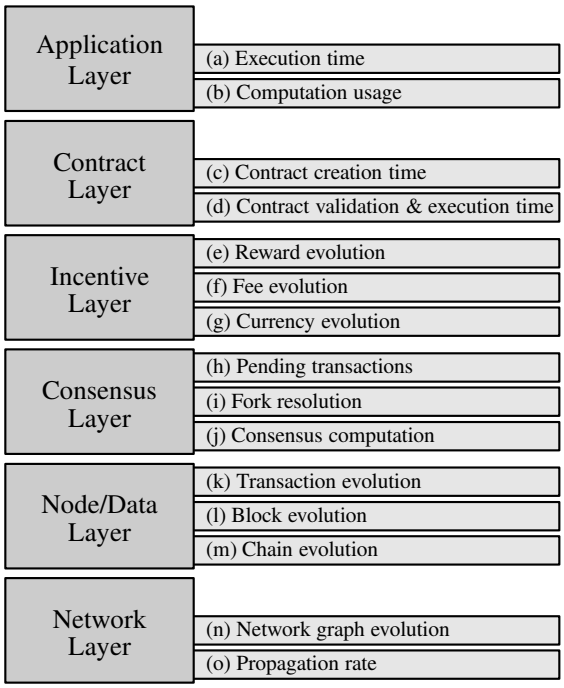


Figure 5.1: Blockchain abstraction layer model & associated metrics denoted by (a) to (o) in the rest of this chapter.

A blockchain is a complex system that can be represented as a six-layer model as depicted in Figure 5.1, which is the adapted model we propose based on the model introduced in [DWC+17]. Then we can define evaluation criteria (metrics) for each of these layers denoted by (a) to (o) in Figure 5.1 (further explained in Section 5.3 and in Chapter 6).

Settings of a blockchain are plentiful and can be divided into two main types: i) 'architecture' parameters such as the total number of nodes, number of administrative nodes (i.e., involved in the consensus process, or in the control of the blockchain) and ii) platform parameters such as the type of consensus, the time between blocks, the block size and so on. This is further detailed in Section 5.3. The configuration of the blockchain with these settings is important since they impact the output performance, consequently carefully choosing the values for these settings is crucial. However, the choice is not always straightforward, especially since - *to the best of our knowledge* - there is no methodology for choosing the optimal blockchain configuration for an application. There are a few works that provide guidance on the choice of the platform depending on the requirements, the application and the characteristics of the platforms. These frameworks that compare and help choose a platform are often limited since they rely on subjective criteria that are based on one's knowledge about the technology, and although they guide the choice of the platform, they do not provide help on the configuration of the latter. Additionally, considering the number of settings and implicitly the number of combinations, the design of experiments is an important phase so as to experiment with a limited and representative amount of configuration samples and reduce the number of runs since the infrastructure required to test and size the blockchain architecture can be very costly (in terms of computational resources, human and financial resources). These settings combination constitutes alternatives that have different impacts on the evaluation criteria. Most importantly, when evaluating with different criteria, it often happens that using the optimum value of a single setting lead to a local optimum for one criteria while negatively impact another. Consequently, the optimal configuration is not the combination of the local optimums, but it is rather the global optimum, which is the configuration that gives the best results with regards to each of the evaluation criteria. This kind of problem is referred to as a multi-response optimisation problem in the literature [MKCA20, KYH08].

In this chapter, we propose a methodology to find the optimal blockchain configuration. This methodology is composed of the three following steps: i) DoE to find a limited and representative sample of configurations that limit the number of runs, ii) the experiments and iii) the ranking of the configurations considering the multi-response optimisation problem. This chapter is organised as follows: Section 5.2 provides some information on the methods that will be used in the chapter, Section 5.3 presents the generic methodology, the methodology is instantiated and the results of the experiments are presented in Section 5.4 and Section 5.5 discusses the limitations of the study and concludes the chapter.

5.2 Background

There are several methods that can be used for each of the steps of our proposed methodology. Multi-response or multi-objective response are often tackled using Multi-Criteria Decision-Making (MCDM) methods in the literature. MCDM methods can be divided in two categories: Multi-Attribute Decison-Making (MADM) and Multi-Objective Decision-Making (MODM) [KRD+16]. MADM methods include methods such as TOPSIS, AHP, PROMETHEE (I & II), VIKOR and MODM methods include techniques such as Genetic Algorithms (GA). In this chapter we consider MADM techniques since they are used to tackle problems with a limited set of alternatives [KRD+16] predetermined by the Design of Experiments step. In this section, we provide background on the methods used in the instantiation of our methodology, even though other MADM could be used.

5.2.1 Taguchi methods

Taguchi methods have been introduced by Genichi Taguchi. These methods have been proposed in the context of quality improvement and the contributions are multiple [MOJH04]:

- Taguchi proposed a loss function
- He introduced Orthogonal Arrays (OAs) to simplify the DoE. Note that he did not invent them, many of them are classical fractional factorial designs [MOJH04], but he simplified their format.
- He introduced robust designs and provided common values for measures called signal to noise (S/N) ratios so as to simplify evaluations and comparisons [OAE18]

Taguchi methods allow for the study of the effect of the parameters on a response (output) and allow for the search of the optimum condition for this response. Note that the optimisation of one response that can degrade another response in the case of a multi-response case [OAE18].

5.2.2 Analytical Hierarchy Process (AHP)

Analytical Hierarchy Process (AHP) is a MCDM/MADM method introduced by Thomas L. Saaty in the 1970s. AHP decomposes the decision in several steps [Saa08]: after defining the problem, the idea is to structure the decision in a hierarchical manner where the top level is the goal of the decision, the level below represents the objectives, the level under represents the criteria and the bottom level is the set of alternatives. Then the next step is to create pairwise comparison matrices to compare elements that are under the same upper-level element. The final step is to use the output of the previous comparisons to weight the priorities in the level directly below, for each element and then for each element in the level below, compute its overall priority by adding its weighted values. This step is to

be reproduced until all the priorities for the alternatives are computed.

Saaty also proposed a scale of values for the pairwise comparisons. These values are used to indicate how important is an element over another. The scale ranges from 1 to 9, 1 means that the elements are of equal importance while 9 means that one element is of extreme importance over the other.

5.2.3 Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS)

Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) is another MCDM/MADM method introduced in the 1980s by Ching-Lai Hwang and Yoon. It has been designed to evaluate and compare different alternatives. TOPSIS works as follow: the first step consists in creating a normalised matrix of alternative and criteria, then the following step consists in calculating the worst and best solution overall. The next step consists in calculating the euclidian distance between each alternative and the best possible solution and the worst possible solution and finally, rank the alternative according to the previous calculation. The idea is that the optimal solution is the closest to the best solution and the farthest from the worst solution [CSTK07].

Note that TOPSIS is often used in combination with AHP such as in [JVSV22, TKK14, THM11, BDFN+16].

5.2.4 Combination of the three previous techniques

Some works in the literature propose and use hybrid methods combining the three aforementioned methods. The main reason for this is to overcome the inability of Taguchi methods to deal with multi-response optimisation as seen in [MKCA20] where the authors propose to use the optimum conditions given by Taguchi methods as alternatives for the AHP and TOPSIS methods in order to optimise the performance of a heat and power system in the field of Fuel Cells.

In [RPRKVK19], the authors use the combination of Taguchi-AHP-TOPSIS to optimise for the electrical discharge machining of lead-induced titanium alloy considering three performance characteristics and four machining process parameters. Similarly, in [PNVNTN22], the authors apply Taguchi-AHP-TOPSIS to optimise two performance criteria considering three parameters in the field of elecrical discharge machining with coated electrodes. The authors also show that the Taguchi-AHP-TOPSIS combination is suitable to solve the multi-response optimisation problems in the field of electrical discharge machining.

To the best of our knowledge, this combination has not been applied in the field of blockchain.

5.2.5 Blockchain & Multi-Criteria Decision-Making (MCDM)

Although, to the best of our knowledge, the combination of Taguchi-AHP-TOPSIS has not been applied in the field of blockchain, there are a few works that have applied MCDM methods for different purposes. Most of the studies using MCDM techniques in the literature in the field of blockchain aim at evaluating or comparing different platforms to help the selection or evaluating and comparing the characteristics of the different platforms, such as S. Nanayakkara *et al.* in [NRP+21] propose a methodology to select a suitable enterprise blockchain based on the Simple Multi Attribute Rating Technique (SMART) MCDM method for the selection. S. Farshidi *et al.* propose a decision support system for platform blockchain selection in which they consider 121 criteria and 28 alternatives. S. Zafar *et al.* in [ZAR21] propose a methodology based on a novel approach (a combination of entropy and CRITIC) to assign weights and apply Weighted Sum Model (WSM), TOPSIS and VIKOR MCDM techniques to rank public blockchain platforms and highlight the suitable ones. H. Tang *et al.* apply TOPSIS and the entropy method (for weight assignment) to evaluate public blockchain platforms with regards to multiple criteria [TSD19].

There are other applications to MCDM methods in the field of blockchain , for example, D. Maček and D. Alagić propose a VECTOR-AHP hybrid method to evaluate compare Bitcoin cryptocurrency security with other internet transaction systems [MA17].

However, to the best of our knowledge, no work addresses the configuration of a blockchain platform.

5.3 Methodology

This section presents the methodology we propose. Figure 5.2 shows the generic methodology and the instantiation of the generic methodology we propose in this chapter.

First, the goal of the study and the methodology is to find and select the optimal blockchain configuration for an application/use-case and a blockchain platform chosen for this application. For this matter, it is required to design the architecture needed for running the experiments and define the global scenario according to the application for which the blockchain platform will be used.

5.3.1 Design of Experiments

This first step - *denoted as the number 1 in Figure 5.2* - aims at defining the experiments so as to limit the number of runs and determine the configurations (parameters combinations) to run. Due to the number of possible configurations and consequently the number of runs and the cost of running such experiments in

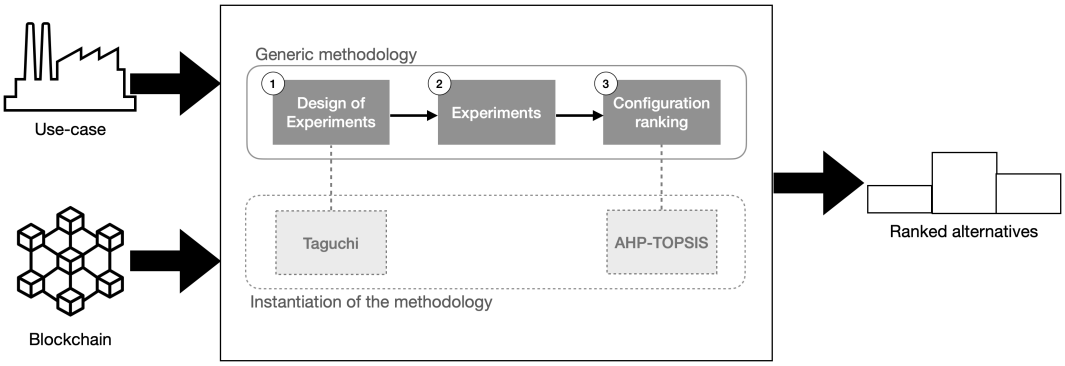


Figure 5.2: Methodology for finding the optimal blockchain configuration

terms of computational resources, human resources and time, it is not envisioned to try all the possible configurations.

The first step in the DoE consists in identifying the inputs and parameters of the application and of the blockchain platform. There are two types of parameters:

1. the 'architecture' parameters: the number of total nodes, the number of nodes taking part in the consensus process, the number of clients submitting transactions to the blockchain

2. the platform parameters: the input transactions (that depends on the application), the block size, the time between blocks, the consensus mechanism used. Note that these also include platform-specific parameters such as the gas price in Ethereum blockchain.

During this step, it is also required to identify the values that can be taken by each of these parameters.

At the same time, evaluation criteria must be defined. In this chapter, we propose the criteria denoted by (a) to (o) in Figure 5.1 described in Table and adapted from the one we describe in Section 6.2 in the next Chapter.

For this DoE step, we propose to use the Taguchi method in this chapter. In the Taguchi method, the parameters are called *factors* and the values taken by the *factors* are called *levels*. There needs to be at least two *levels*, for example it can be minimum value and maximum value, but there can be more.

When using the Taguchi method, it is required to identify the Orthogonal Array (OA) to use. OAs are predefined by Taguchi, and the selection of an OA is based

Table 5.1: Proposed evaluation criteria

Layer	Criterion	Description
Application	(a) Execution time	Time required for the platform to process the desired volume of information
	(b) Computation usage	Usage of computing resources (i.e., CPU, memory, ...)
Contract	(c) Contract creation time	Time required for the platform to generate the contract(s)
	(d) Contract validation and execution time	Time required for the platform to validate and execute the contract(s)
Incentive	(e) Reward evolution	The amount of cryptocurrency distributed from the consensus process
	(f) Fee evolution	Evolution of the reward fees that nodes offer to miners to incentivise them to process their transaction(s)
	(g) Currency evolution	Amount of cryptocurrency generated
Consensus	(h) Pending transactions	The number of transactions that are waiting to be validated
	(i) Fork resolution	The number of forks happening within the chain and the stale rate
	(j) Consensus computation	Computational effort required to validate transactions and blocks
Node/data	(k) Transaction evolution	The number of transactions generated over the course of the run
	(l) Block evolution	Regroups the number of valid blocks generated over the course of the run (l_1), average time taken by a block to be validated (l_2), block size (l_3)
	(m) Chain evolution	Length of the chain
Application	(n) Network graph evolution	Regroups network graph metrics such as Clustering Coefficient, Mean Geodesic Distance and Diameter
	(o) Throughput	Number of valid transactions per second

on the number of *factors* (denoted as X in Figure 5.3) and the number of *levels* (denoted as n in Figure 5.3) identified. An OA is described as Ly (n^X), where y is the number of runs, n is the number of *levels* and X is the number of *levels*. As an example, L4 (2^3) is a OA composed of 4 runs, 3 *factors* of 2 *levels* each. The OA defines the configurations (parameters combinations) to run and will allow for the evaluation of the effect of the parameters on a response (criterion).

5.3.2 Experiments

The second step - *denoted as number 2 in Figure 5.2* - is to run the experiments according to the DoE and develop the code(s) required to retrieve the results for the evaluation criteria selected.

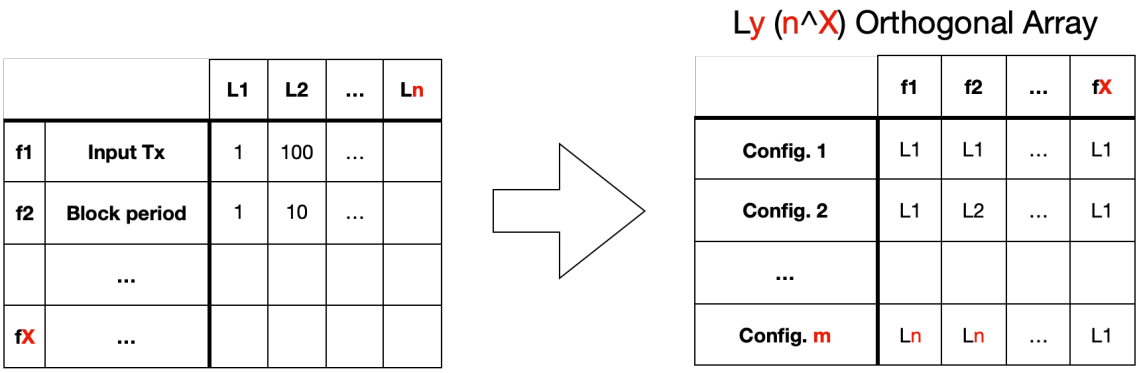


Figure 5.3: Taguchi Orthogonal Array illustration

After gathering the results, an optional step when using Taguchi methods is to study the optimum conditions for each response/criteria individually by conducting Taguchi's signal to noise (S/N) ratios analysis and also study the effects and contributions of the parameters on the response, as explained and detailed in [MKCA20].

5.3.3 Configuration ranking

At this stage of the methodology, the experiments are done and the results for each configuration (alternative) with regards to the defined criteria are gathered. As mentioned in the Introduction section (5.1) and as seen in the results, using the optimum value for one criteria can impact negatively another criteria (multi-response optimisation problem). As an example, when increasing the throughput of the blockchain system, the resource consumption will automatically increase, as a consequence it is difficult to find the optimal configuration that allows for the maximum throughput while maintaining a minimum resource consumption. And it gets more difficult as the number of parameters and criteria increases. In such a case, the configuration ranking step - *denoted as number 3 in Figure 5.2* - should be able to tackle the multi-response optimisation problem.

In this chapter, we propose to use AHP for assigning weights to the alternatives/configurations and TOPSIS to rank them.

The structure of the decision should be as depicted in Figure 5.4. The level 0 (L0) is the goal of the decision, namely finding the optimal blockchain configuration. Then at the level below (L1) should appear the layers of the blockchain for which criteria have been selected, the level below (L2) contains the evaluation

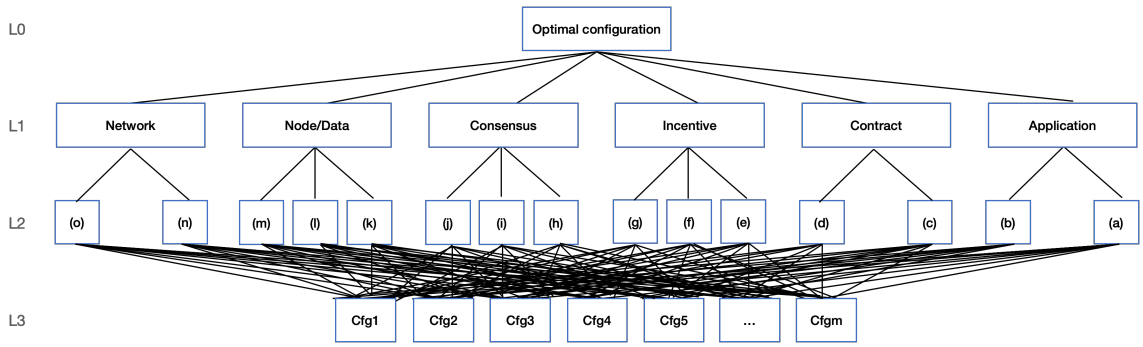


Figure 5.4: Optimal configuration decision-making structure

criteria and the lowest level (L3) is composed of the alternatives (configurations given by Taguchi's OA). Note that we choose to keep the alternatives resulting of Taguchi's OA because we want to give the option to an expert to change the priorities/importance of some criteria over the other. It is possible optimise the number of alternatives by keeping only the optimum conditions as alternatives, as in [MKCA20]. The final tree can change depending on the criteria selected.

The weights for the L1 level are preference-based and are calculated as follow:

1. The pairwise comparison matrix P of size $m \times m$, with m being the number of L1 criteria or layers here, should be created as in 5.1. w_{ij} are the preferences specified using Saaty's scale. As an example, w_{1m} is the preference of criterion C_1 over C_m, and if $w_{1m} = 1$, it means that C_1 and C_m are of equal importance, however, if $w_{1m} = 9$, it means that C_1 is extremely more important than C_m.

$$\mathbf{P} = \begin{array}{c} \\ C_1 \\ \vdots \\ C_m \end{array} \begin{array}{c} \begin{array}{ccc} C_1 & \cdots & C_m \end{array} \\ \begin{bmatrix} w_{11} & \cdots & w_{1m} \\ \vdots & \ddots & \vdots \\ w_{m1} & \cdots & w_{mm} \end{bmatrix} \end{array} \tag{5.1}$$

2. The weights for each criterion W_{C_i} are calculated using equation 5.2 and W_C is the resulting weight vector composed of the weights for each criterion.

$$W_{C_i} = \frac{\sum_{j=1}^{m} w_{ij}}{\sum_{k=1}^{m}\sum_{j=1}^{m} w_{kj}}, \qquad w_{ij} = \begin{cases} 1 & i = j \\ \frac{1}{w_{ij}} & i \neq j \end{cases} \tag{5.2}$$
$$W_C = [W_{C_1}, \ldots, W_{C_i}, \ldots, W_{C_m}]$$

This should be conducted in the same way for the L2 level, except that there should be as many matrices as the number of elements in L1 (m). For example, in this case, there should be six matrices, one for each L1 criteria, for which the size is the number of criteria under the same L1 element: for Network, the weight matrix P is of size 2×2 since there are two criteria under Network.

For the L3 level, however, it is slightly different since the pairwise comparison matrices of this (alternatives) level are score-based, and are created from the results obtained out of the experiments. The pairwise comparison matrix S of size $z \times z$, z being the number of alternatives, is then created as in 5.3, with A being the alternatives and $C_i(A_z)$ being the result of alternative A_z with regards to the C_i criterion:

$$\mathbf{S} = \begin{array}{c} \\ A_1 \\ \vdots \\ A_z \end{array} \begin{array}{c} \begin{array}{ccc} A_1 & \cdots & A_z \end{array} \\ \begin{bmatrix} 1 & \cdots & \frac{C_i(A_1)}{C_i(A_z)} \\ \vdots & \ddots & \vdots \\ \frac{C_i(A_z)}{C_i(A_1)} & \cdots & 1 \end{bmatrix} \end{array} \tag{5.3}$$

Note that the previous S matrix is for the case of a 'benefit' criteria, which means that the alternative that has the maximum value with regards to this criterion is the best. In the case of a 'cost' criteria, meaning that we want to minimise the value, the matrix is calculated as in 5.4.

$$\mathbf{S_{cost}} = \begin{array}{c} \\ A_1 \\ \vdots \\ A_z \end{array} \begin{array}{c} \begin{array}{ccc} A_1 & \cdots & A_z \end{array} \\ \begin{bmatrix} 1 & \cdots & \frac{1}{\frac{C_i(A_1)}{C_i(A_z)}} \\ \vdots & \ddots & \vdots \\ \frac{1}{\frac{C_i(A_z)}{C_i(A_1)}} & \cdots & 1 \end{bmatrix} \end{array} \tag{5.4}$$

Then, the scores of each alternative with regards to the criteria can be computed using equation 5.5, where $W_{C_i}^{A_l}$ is the score of alternative A_l with regards to the C_i criterion and $W_{C_i}^{A}$ the vector of size n (being the number of alternatives) containing the scores of all the alternatives with regards to the C_i criterion:

$$W_{C_i}^{A_l} = \frac{\sum_{j=1}^{z} S_{ij}}{\sum_{k=1}^{z}\sum_{j=1}^{z} S_{kj}}, \qquad S_{ij} = \begin{cases} 1 & i = j \\ \frac{1}{S_{ij}} & i \neq j \end{cases} \tag{5.5}$$

$$W_{C_i}^{A} = [W_{C_i}^{A_1}, \ldots, W_{C_i}^{A_z}]$$

Note that there should be as many $W_{C_i}^{A}$ as criteria.

Once the scores of each alternative with regards to the criteria are computed, it is possible to apply the TOPSIS method. The input of TOPSIS is then $GW_{C_i}^{A_l}$ and is calculated as in equation 5.6 where $W_{C_i}^{A_l}$ is the score of alternative A_l with regards to the C_i criterion, $W_{C_i}^{L2}$ is the weight of each criterion at level L2 and $W_{C_i}^{L3}$ is the weight of each criterion at level L3.

$$GW_{C_i}^{A_l} = W_{C_i}^{A_l} \cdot W_{C_i}^{L2} \cdot W_{C_i}^{L1} \tag{5.6}$$

Then the best (d_i^+) and worst (d_i^-) solutions have to be computed as in equations 5.7 and 5.8.

$$d_i^+ = \max_{l=1,\ldots,z} \left(GW_{C_i}^{A_l}\right) \tag{5.7}$$

$$d_i^- = \min_{l=1,\ldots,z} \left(GW_{C_i}^{A_l}\right) \tag{5.8}$$

After this, $D^+(A_l)$ and $D^-(A_l)$, which are respectively the geometric distance of alternative A_l to the best and worst solutions have to be computed using the equations 5.9 and 5.10.

$$D^+(A_l) = \sqrt{\sum_i \left(GW_{C_i}^{A_l} - d_i^+\right)^2} \qquad l = 1, \ldots, z \tag{5.9}$$

$$D^-(A_l) = \sqrt{\sum_i \left(GW_{C_i}^{A_l} - d_i^-\right)^2} \qquad l = 1, \ldots, z \tag{5.10}$$

Finally, it is possible to compute $R(A_l)$, which is the global rank score of alternative A_l using equation 5.11. Once this is computed for all the alternatives, the alternatives can be ranked from best to worst according to their global rank score.

$$R(A_l) = \frac{D^-(A_l)}{D^+(A_l) + D^-(A_l)} \qquad l = 1, \ldots, z \tag{5.11}$$

5.4 Implementation & Results

This section presents the instantiation of the methodology for our case study. In this chapter, we want to find the optimal blockchain configuration for storing of process and production data for traceability purposes. The platform selected in this chapter is Ethereum with Clique Proof-of-Authority consensus mechanism.

5.4.1 Experimental setup

Production data is collected through a wrapper that contains an OPC-UA client and crafts and sends transactions containing the data to store - *by calling a function of a smart contract previously deployed on the blockchain* - to Go-Ethereum (Geth) nodes that are deployed in docker containers on three servers. For these experiments, we choose to expose only one Geth node, meaning that there is only one node getting transactions from the wrapper and broadcasting them to the Ethereum network. OPC-UA is configured in request-response communication mode so as to be able to control the input throughput of the blockchain by controlling the request period.

5.4.2 DoE - Parameters, criteria and configurations

For our Ethereum implementation, we identified five parameters or factors:

- $f2$ - Block period (time between blocks (in seconds))
- $f3$ - Number of OPC-UA variables to retrieve in each request
- $f4$ - Request period (time between OPC-UA requests (in seconds))
- $f5$ - Number of non-signer nodes
- $f6$ - Number of signing nodes (miners)

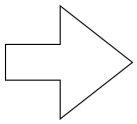
For these parameters we identified four levels. These levels for are described in Table 5.2. We choose to conduct the study with four level to better capture the influence of the parameters.

Table 5.2: Factors and associated levels

	L1	L2	L3	L4
$f2$	0	20	60	300
$f3$	2	20	60	120
$f4$	5	10	30	45
$f5$	1	2	3	4
$f6$	1	2	3	4

With five factors and four levels, Taguchi gives a L16 (4^5) array, containing 16 configurations to run. Figure 5.5 shows the L16 array filled with our parameters' values. Configurations 1 to 16 are the configurations to run as experiments in the next step.

	L1	L2	L3	L4
f1	0	20	60	300
f2	2	20	60	120
f3	5	10	30	45
f4	1	2	3	4
f5	1	2	3	4



L16 (4^5) Orthogonal Array

	f1	f2	f3	f4	f5
Config. 1	0	2	5	1	1
Config. 2	0	20	10	2	2
Config. 3	0	60	30	3	3
Config. 4	0	120	45	4	4
Config. 5	20	2	10	3	4
Config. 6	20	20	5	4	3
Config. 7	20	60	45	1	2
Config. 8	20	120	30	2	1
Config. 9	60	2	30	4	2
Config. 10	60	20	45	3	1
Config. 11	60	60	5	2	4
Config. 12	60	120	10	1	3
Config. 13	300	2	45	2	3
Config. 14	300	20	30	1	4
Config. 15	300	60	10	4	1
Config. 16	300	120	5	3	2

Figure 5.5: Taguchi L16 Orthogonal Array with configurations

Then, for the evaluation of the blockchain configurations, we use five of the criteria proposed in our methodology in the previous section, namely:

- (b_1) - CPU consumption
- (b_2) - Memory consumption
- (k) - Number of valid transactions processed at the end of the experimental run
- (l_1) - Number of valid blocks processed at the end of the experimental run
- (l_2) - Average time for a block to be validated as part of the blockchain

5.4.3 Experiments

Each of the 16 configurations are run for one hour. Table 5.3 presents the results

For the (b_1) and (b_2) metrics, Table 5.3 only reports the consumption of the exposed node sending the transactions since the consumption of the other nodes is not significant in comparison.

Additionally, there is no result for configurations 2, 3, 4 and 11. Configurations 2, 3 and 4 all have the block period set to 0s, which means that the signer will sign

Table 5.3: Results for each configuration with regards to the evaluation criteria

	(l_1)	(k)	(l_2) (in s)	(b_1) (%)	(b_2) (%)
Config. 1	1438	1438	17523	0.75	0.12
Config. 5	180	718	146065	1	0.11
Config. 6	181	14380	142675	7.1	0.3
Config. 7	180	4740	142126	2.48	0.27
Config. 8	178	14380	140000	11.59	0.48
Config. 9	60	238	424035	0.73	0.05
Config. 10	60	1580	419999	2.69	0.16
Config. 12	61	43080	423170	197.57	1.07
Config. 13	13	158	2100263	1.34	0.05
Config. 14	13	2380	2107206	6.46	0.2
Config. 15	13	21540	2099976	181.47	1.54
Config. 16	14	31718	2105008	318.79	2.52

the block as soon as he receives a transaction, this does not work when there is more than one signer since no block is created. Indeed, it seems like they are not able to determine which signer will sign when they all receive a transaction in this case. Configuration 11 makes the signer crash every time we run the experiment. It is important to note also that due to the load induced by configurations 15 and 16, it happens that the experiments fail because of crashes. In the rest of the study, 12 configurations are considered.

Finally, we can observe some deviations compared to the theoretical results, for example, in the results with configuration 6, 20 variables are requested every 5 seconds, there is one block every 20 seconds and the experiment lasts 60 minutes so theoretically, there should be 180 blocks and 14400 transactions and around 80 transactions per block, while in practice there are 181 blocks and 14380 transactions. However we could observe that the first few blocks are not full when they are signed, there is a transitional period at the beginning of the experiment.

5.4.4 Configuration ranking

Now that the alternatives/configurations are defined by Taguchi method and the experiments and the experiments are done, the configuration ranking phase can start. Figure 5.6 presents the final AHP tree obtained. This tree only contains the layers and criteria we consider as well as only the alternatives for which we have results regarding all the criteria.

According to the structure of the decision, we apply the equations for AHP and TOPSIS presented in the previous section to compute the ranking of the alternatives. All the detailed computations can be found in Appendix (8) in Section 8.1

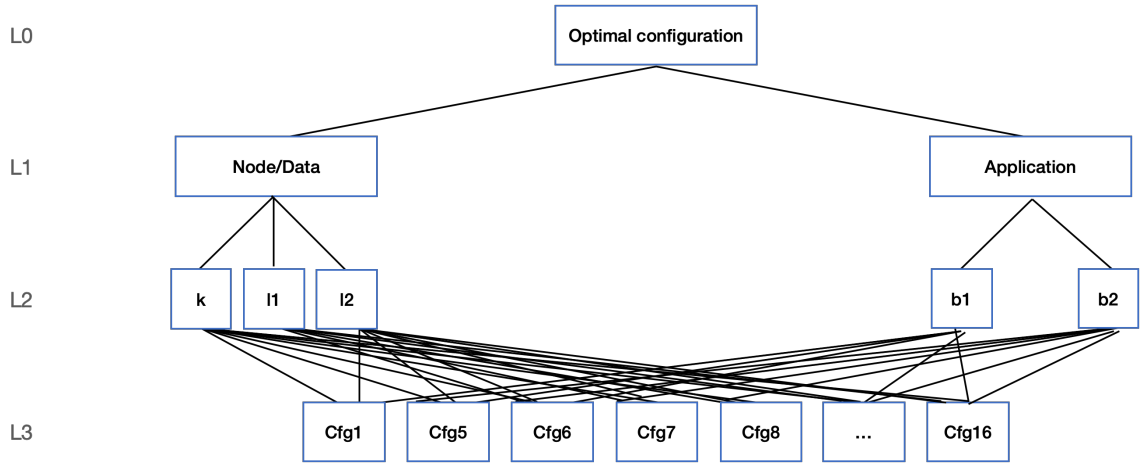


Figure 5.6: Optimal configuration decision-making structure to weight and rank our 12 alternatives

and 8.2. Vector 5.12 gives the final score of each alternative after all the AHP and TOPSIS computations.

$$R(A) = \begin{bmatrix} 0.5703 \\ 0.1938 \\ 0.1162 \\ 0.1182 \\ 0.1036 \\ 0.2607 \\ 0.0907 \\ 0.1259 \\ 0.1924 \\ 0.0525 \\ 0.0588 \\ 0.0818 \end{bmatrix} \tag{5.12}$$

The top 3 configurations when the preferences are all set to 1 are then, in order: Configuration 1, configuration 9 and configuration 5, as depicted in Figure 5.7.

5.4.5 Influence of the AHP preferences on the final ranking

In the previous application of the methodology, all the preferences in the AHP matrices were set to 1 on Saaty's scale. But the main advantage of using AHP is to

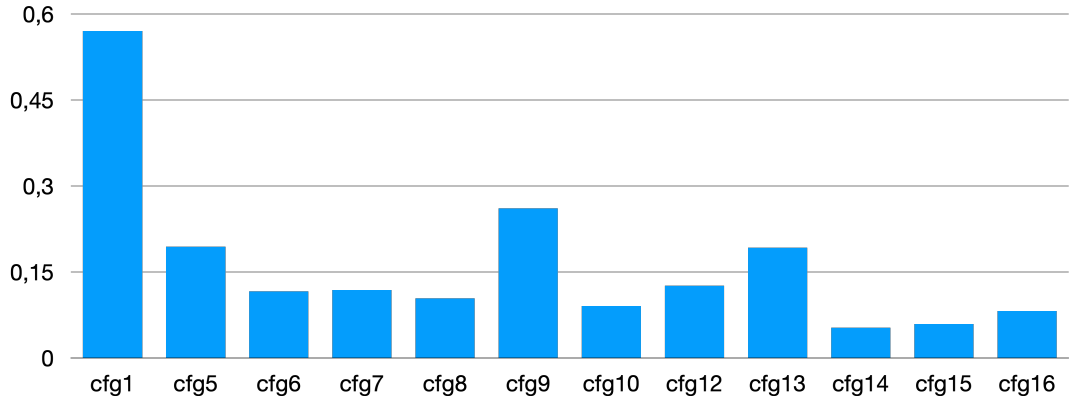


Figure 5.7: Final scores of the alternatives when all the preferences are set to 1

set preferences on criteria over the others. As an example, if the expert considers the resource consumption as a more important criteria, he can directly specify it in the L1 matrix. It is also possible to apply different preferences on specific criteria in the L2 matrices rather than L1 which is a category of criteria. Let us demonstrate the impact of this in the example of a expert willing to minimise the resource consumption. The resource consumption criteria are in the application layer. The P_{L1} matrix is then described in 5.13 and the weight vector W_{L1} is computed as in 5.14.

$$\mathbf{P_{L1}} = \begin{array}{c} \\ Node/Data \\ Appli. \end{array} \begin{array}{c} \begin{array}{cc} Node/Data & Appli. \end{array} \\ \begin{bmatrix} 1 & \frac{1}{9} \\ 9 & 1 \end{bmatrix} \end{array} \tag{5.13}$$

$$\begin{aligned} W_{Node/Data} &= \frac{1+0.1}{1+0.1+9+1} = 0.1 \\ W_{Appli.} &= \frac{9+1}{1+0.1+9+1} = 0.9 \\ W_{L1} &= [W_{Node/Data} \quad W_{Appli.}] = [0.1 \quad 0.9] \end{aligned} \tag{5.14}$$

The remaining of the AHP computations then stay the same as before. However, the input matrix of TOPSIS GW_C^A changes to integrate the modifications of W_{L1} and the TOPSIS method has to be applied considering the new matrix. The

$R(A)$ vector in 5.15 presents the final scores of the alternatives when the criteria of the Application layer are preferred over the Node/Data layer criteria with a preference of 9 on Saaty's cale.

$$R(A) = \begin{bmatrix} 0.3444 \\ 0.2417 \\ 0.0681 \\ 0.1049 \\ 0.0479 \\ 0.3973 \\ 0.1186 \\ 0.0314 \\ 0.3064 \\ 0.0733 \\ 0.0169 \\ 0.0185 \end{bmatrix} \tag{5.15}$$

In this case, the top 3 configurations are configuration 9, configuration 1 and configuration 5, in this order, as depicted in green in Figure 5.8 (the blue graph is when all the preferences are set to 1 for comparison). This shows that the preferences definitely have an impact on the ranking of the configurations and that they need to be cautiously defined according to the responses one wants to optimise.

5.5 Limitations & conclusion

5.5.1 Limitations

There are some limitations to the study. First, regarding the methodology itself, it has been instantiated using one set of methods for the moment.

Then, there are limitations regarding the experiments and the results. Indeed, we collect data on one machine, and send the data to the blockchain through one client, so there is no indication regarding the performance of the blockchain when the scale increases. Additionally, there is no result for configuration 2, 3, 4 and 11, which means that there is sometimes only one configuration for evaluating a precise value for a parameter, for example for a block period ($f2$) of 0s, only configuration 1 has results. This can be part of the reason why configuration 1 is always among the best configurations. Note that this is not a limitation of the methodology itself. Finally, still on the topic of experiments, storing data in a smart contract might not be the optimal way even though it is the most straightforward way.

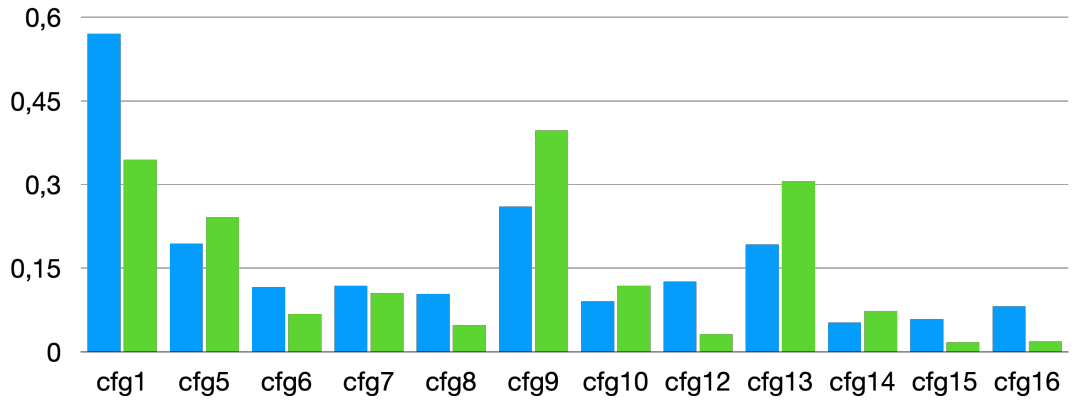


Figure 5.8: Comparison of the scores when all preferences are set to 1 (blue) and when the Application layer criteria are preferred to the Node/Data layer criteria with a preference of 9 (green)

5.5.2 Conclusion & perspectives

Blockchain systems are complex systems, consequently, there are numerous factors that impact the performance of such systems. One of these factors is the configuration of the input parameters of the blockchain. Indeed, there a a significant amount of parameters to a blockchain system and the configuration is not straightforward and there is a lack of guidance on this subject in the literature. For this matter, this chapter proposes a generic methodology for selecting an optimal blockchain configuration for an application. This methodology consists in three steps, namely the design of experiments, the experiments (where one can the blockchain implement a platform for the desired application/use-case) and the configuration ranking based on the previous experiments. This methodology has been instantiated using Taguchi DoE method, the experiments were conducted on a private Ethereum platform for storing production and process data and the configurations were evaluated and ranked using the AHP and TOPSIS methods.

Although we demonstrate the applicability of our proposed methodology in this chapter, there are still some limitations to the methodology and to the implementation and experiments as previously mentioned. It could be interesting to compare the output of the same methodology instantiated with another set of methods. In addition, regarding the experiments, it could be interesting to increase the number of data sources as well as the number of nodes sending transactions to further

assess the suitability of the blockchain for traceability purposes. With a bigger volume of data to store, it might also be required to investigate the use of proper data storage systems on blockchain such as Swarm in Ethereum for example.

Finally, when running the experiments, we could notice the difficulty and the time required to change the parameters and implement each configuration. Also, scaling up the experiments in terms of nodes is not always possible since one might not have the resource for that, for example for us, scaling up means adding docker containers on the same servers as the other since we do not have more servers. Doing so could give an idea of the behaviour of the system when the scale increases, but it is limited. We believe that simulation/emulation could help overcome some of the limitations, especially the limitations concerning the experiments. As a matter of fact, simulation could ease the process of changing the configuration, and most importantly, it could help experiment with a higher number of nodes and a higher number of data sources. This is what we initially started investigating by exploring blockchain simulators, but since there is a lack of exhaustive blockchain simulator that covers all the layers presented in the Introduction section (in Figure 5.1), we propose BlockPerf, a blockchain simulation/emulation framework, in the next chapter.

6

BlockPerf: A hybrid blockchain emulator/simulator framework

Contents

6.1 Introduction

Blockchain is increasingly applied to all sectors of our daily life, spanning from financial applications [MSA19, AJS19] to industrial ones [AJM19, BTP+20]. Blockchain technology is a type of Distributed Ledger Technology (DLT) that uses a ledger stored in a distributed manner and shared among its participants in the network [ZXD+18].

Decentralization, consistency, anonymity and traceability are its intrinsic features making it an interesting technology for many applications in which such aspects must be tackled. However, it is never an easy task for researchers, developers, and practitioners to decide what blockchain technologies) they should select/implement, as application requirements may significantly vary from one application to another (e.g., in terms of what data should be stored, the number of transactions to be performed, etc.), without speaking about the multiple constraints of networking, computing power and communication that the application may face [FGKM20]. Several blockchain performance assessment frameworks have been proposed in the literature to overcome this difficulty, as presented in [MSH19, GKLS20, PRLT20a], but yet they often focus on functional aspects (e.g., type of consensus, support of smart contracts) and consider fixed performance values (collected from the literature) for dynamic criteria such as throughput, latency, block validation time, etc. The reason for this is that it is complex to formalize and model the performance of such dynamic criteria [FGKM20, TWL+20], as they depend on a wide range of parameters and inter-dependent layers, as described in the six-layer model presented in Figure 6.1. Looking at significant blockchain technologies such as Bitcoin [Nak08], Ethereum [eth], or Hyperledger Fabric [ABB+18], layers differ from one framework to another, and even inside a given framework, different parameter configurations are made possible. Therefore, it is essential to evaluate the blockchain performance based on both real-life and/or simulated environments, without which the blockchain selection process cannot be optimal.

When looking at papers that experimentally evaluate blockchain performance, the majority requires the implementation of the whole system (i.e., using a large set of computers/machines) [SOK+20], which also applies to well-known practical/benchmarking tools such as BlockBench [DWC+17] or Hyperledger Caliper. Such approaches/tools incur high costs for deployment, lack of scalability (e.g., to carry out large-scale experiments) and modularity. On the other hand, simulators can help to deploy and test blockchain technologies in large-scale infrastructure settings. To date, several blockchain simulators exist (e.g., BlockSim [FC19], PeerSim [MJ09], Shadow [MJ15], Vibes [SZJ17], etc.), but they are often limited in several respects. Recent literature reviews of existing blockchain simulation tools [PGF21, HKZ+21] point out the fact that those tools often limit themselves to evaluate part of the blockchain system (i.e., they fail in covering all layers and

associated performance metrics (a) to (o) emphasized in Figure 6.1). This is particularly stressed by R. Paulavicius *et al.* [PGF21] in their systematic review and empirical survey of blockchain simulators, in which the authors conclude that *"there is no 'one-size-fits-all' Proof-of-Work blockchain simulator that is able to accurately simulate all the layers"*. To overcome such a limitation, a new hybrid blockchain emulator/simulator called *BlockPerf* is proposed, which extends the BlockSim simulator proposed by Faria *et al.* [FC19]. One of the main improvement lies in the fact that BlockPerf relies on real network infrastructure (at the Network layer) while simulating the upper layers, leading to more realistic results than existing simulators.

State-of-the-art simulators and the extent to which they cover the six-layer model and associated performance metrics are reviewed and discussed in section 6.2. The architectural design of BlockPerf, and how it extends BlockSim, is presented in section 6.3. In section 6.4, a performance comparison analysis between BlockPerf and BlockSim is carried out based on a real-life (benchmarking) bitcoin scenario; the conclusions and BlockPerf limitations being discussed in section 6.5.

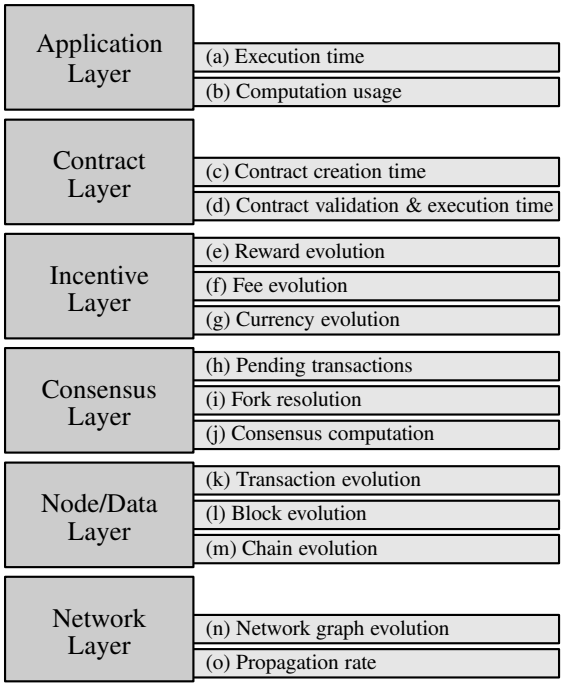


Figure 6.1: Blockchain abstraction layer model & associated metrics denoted by (a) to (o) in the rest of this paper.

6.2 Background and Related Work

As previously discussed, evaluating a blockchain technology or platform turns out to be a complex process due to the various inter-dependent layers and associated parameters. To better explain this complexity, a slightly adapted version of the model introduced in [DWC+17] is considered in this paper, which corresponds to the six-layer model given in Figure 6.1. Although one may find other blockchain abstraction models in the literature, as the ones proposed by the ITU and ISO standardization bodies [itu, ISO20], we adopted this six-layer model as it covers most aspects of a blockchain (DLT) platform, and it is straightforward to understand. Nonetheless, to allow readers to understand to what extent this model covers the ongoing ITU and ISO blockchain standard initiatives, and *vice-versa*, we provide a summary table in Table 6.1. Considering this model, sections 6.2.1 to 6.2.6 provide the necessary background regarding each of these layers, starting from top (Application) to bottom (Network), by detailing the key metrics – *(a) to (o)* – that a simulator should allow for measuring/tracking throughout a simulation run. Section 6.2.7 then discusses the extent to which existing blockchain simulators cover those layers and metrics.

Table 6.1: Extent to which the six-layer model introduced in Figure 6.1 covers the ITU and ISO blockchain standard initiatives, and *vice-versa*

	Layer Focus					
Standards	Application	Contract	Incentive	Consensus	Node/Data	Network
ITU-T SG16 Q22 F.751.1	✔	✔		✔		✔
ITU-T SG16 Q22 F.751.2		✔		✔	✔	✔
ISO/DIS 22739	✔	✔	✔	✔	✔	✔
ISO/WD TS 23258		✔		✔	✔	✔
ISO/CD 23257				✔	✔	✔
ISO/CD TR 23576					✔	
ISO/AWI TS 23259		✔				

6.2.1 Application Layer

This layer manages the user interface, APIs (Application Programming Interface), and computational resources (e.g., needed for blockchain element storage, wallet creation, etc.). In many blockchains, there is the possibility to configure a node to run as a full node (storing a local copy of all transactions and blocks) or as a lightweight node (in charge of creating transactions and sending them to full

nodes for validation purposes). From a simulation (or emulation[1]) perspective, the following metrics should be measurable at this layer:

(a) *Execution time:* it refers to whether the simulator keeps track of the time needed to run the simulation;

(b) *Computational resource usage:* it refers to whether the simulator keeps track of the resource usage evolution throughout the (simulation) run, which includes CPU of each node, swap space, storage capacity, etc.:

6.2.2 Contract Layer

Blockchain systems consist of scripts, also referred to as “smart contracts", that run when predetermined conditions are met (e.g., used to automate the execution of an agreement so that all participants can be immediately certain of the outcome). At this layer, the following metrics should be measurable:

(c) *Contract creation time:* it refers to whether the simulator keeps track of the time needed to generate the different contracts (which can be of different sizes) over the simulation;

(d) *Contract validation & execution time:* it refers to whether the simulator keeps track of the time needed to validate and execute the different contracts of the simulation scenario.

6.2.3 Incentive Layer

Participation incentive means the way how honest behavior is incentivized and dishonest discouraged. Incentives can be in the form of transaction fees and/or rewards [MSA19]. This decision affects the implemented consensus algorithm (introduced in the next model layer) and, respectively, is affected by the selected algorithm. At this layer, the following metrics should be measurable:

(e) *Reward evolution:* it refers to whether the simulator keeps track of the amount of cryptocurrencies distributed from the consensus process (e.g., leader election or mining) over the simulation;

(f) *Fee evolution:* it refers to whether the simulator keeps track of the reward fees that client nodes offer to miners to incentivize them to process their transaction(s).

[1]Simulation/Simulator and Emulation/Emulator are interchangeably used in the rest of the paper.

(g) *Currency evolution:* it refers to whether the simulator keeps track of the currency generation rate, which evolves differently according to the implemented blockchain (e.g., in bitcoin or Ethereum, it evolves along with the hashing difficulty).

6.2.4 Consensus Layer

Consensus protocols are needed to validate the data to prevent and remove any duplicated entry and/or fraud [BMZ18, MXZ+17]. The type of blockchain to be implemented (public, private, consortium) influence profoundly the type of consensus protocol to be used, the most well-known being Proof-of-Work (PoW), Proof-of-Stake (PoA), or still Practical Byzantine fault tolerance (PBFT). At this layer, the following metrics should be measurable:

(h) *Pending transactions:* it refers to whether the simulator keeps track of the number of transactions, over time, that are waiting to be confirmed (such a waiting area is called "Mempool" in Bitcoin, or sometimes called transaction queues);

(i) *Fork resolution:* it refers to whether the simulator keeps track of (i_1) the number of forks that appear within the chain; (i_2) the stale rate (i.e., block discarded) throughout the simulation;

(j) *Consensus computation:* it refers to whether the simulator keeps track of the collective (or individual) computation effort required to validate transactions and blocks.

6.2.5 Node/Data Layer

The node (or data) layer is responsible for structuring the data before appending it to blocks, whose structure usually includes information such as previous block hash, Merkle root, time, bits, etc. At this layer, the following metrics should be measurable:

(k) *Transaction evolution:* it refers to whether the simulator keeps track of the number of transactions generated per day (k_1) and whether the simulated transactions match the real-life data structure (k_2). Unlike k_1, k_2 is not a quantifiable metric but rather a boolean metric that states whether the simulator generates transaction following the real-world blockchain specification;

(l) *Block evolution:* it refers to whether the simulator keeps track of (l_1) the number of blocks that are validated, mined and accepted as part of the longest chain; (l_2) the (average) time taken by each block to be validated; (l_3) the block sizes, which depend on the size of the transactions they include;

and finally (l_4) the number of transactions included, on average, within a block.

(m) *Chain evolution:* it refers to whether the simulator keeps track of the length of the chain over time (i.e., the number of blocks that form the longest chain), which is a good indicator of the load a new node would have to compute if it joins the network at a given point in time.

6.2.6 Network Layer

Blockchain is a pure P2P network, which is actually an overlay network [WHH+19] for distributed object storing, searching,and sharing (e.g., Ethereum relies on the Kademlia P2P protocol [ZHLL19, WHH+19]). At this layer, the following metrics should be measurable:

(n) *Network graph evolution* it refers to whether the simulation accurately follows the P2P protocol overlay network specifications (e.g., support for adding and discovering a node at any time) and keeps track of the network (node) evolution using network graph metrics such as Clustering Coefficient, Mean Geodesic Distance, and Diameter [AB02, JW18];

(o) *Throughput:* it refers to whether the simulator keeps track of the number of valid transactions per second (Tx/s) that have been incorporated as part of a valid block within the longest chain. The transactions that are considered here are the ones that reach the majority of nodes within the network depending on the underlying consensus (i.e., $\geq$ 50% for PoW, $\geq$ 66% for PBFT, etc.);

6.2.7 Related Work and Discussion

As of today, there are several blockchain simulators found in the literature, a summary of which is in Table 6.2. This table highlights what layers and associated metrics those simulators cover. Note that a simulator may, in some cases, cover a given layer but without necessarily providing as output a performance metric. This means that either a modification of the source code or post-processing treatments are required in order to compute the desired metric. For example, in BlockSim, the authors claim in their paper [AvM19] that the simulator keeps track of the fee evolution, however, after analyzing it, we realized that it is not possible to retrieve it without modifying part of the code. Another example is HIVE [hiv] that emulates the behavior of smart contracts within the Ethereum environment, however, we found that it does not allow for analyzing the (h) metric (i.e., pending transactions). To make a distinction between simulators that make available a performance metric without requiring any code modification or post-processing,

and the ones that require a modification/post-processing to obtain the metric, the following two symbols are respectively used in Table 6.2: ● (does not require any code modification and/or post-processing) and ◐ (does require a code modification and/or post-processing).

Table 6.2: Literature comparison - ● symbol indicates that the simulator makes available the performance metric without any code modification and/or post-processing, while ◐ symbol indicates that such a modification and/or post-processing is needed to obtained the desired performance metric

	Application		**Contract**		**Incentive**			**Consensus**			**Node/Data**			**Network**	
Framework	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)	(j)	(k)	(l)	(m)	(n)	(o)
DLSF [dsl]	◐	◐			●			●	◐	◐	●$_{k1,k2}$	●$_{l1,l2,l3,l4}$	◐		◐
Shadow [MJ15]	●	●			◐				●		●$_{k1}$	◐$_{l2,l4}$	◐		●
Wang et al. [WCY+18]	◐	●			●			●			●$_{k1}$	●$_{l2,l4}$			
BlockSim-Net [ABK+20]	◐	●					◐		●	◐	◐$_{k1,k2}$	●$_{l1,l2,l3}$	◐		◐
eVibes Plasma [evi]	◐		◐	●					●		◐	◐	◐		◐
Vibes [SZJ17]	●							◐			●$_{k1}$	●$_{l1,l2,l4}$	◐		●
eVibes [DNJ18]	◐			●					●		●$_{k1}$	●$_{l1,l2,l4}$			●
BlockSim [AvM19, FC19]	●	◐				◐			●		●$_{k1}$	●$_{l1,l2}$	◐		●
Goswami [Gos17]	◐								◐		●$_{k1}$	●$_{l2,l4}$			
Bitcoin-Simulator [GKW+16]	◐						◐		◐		●$_{k1}$	●$_{l1,l2,l3,l4}$	◐		●
SIMBA [FMF20]	◐								◐	◐	●$_{k1}$	●$_{l2,l3,l4}$	◐		◐
Chin et al. [CYT20]	◐				◐		●		●		●$_{k1}$	●$_{l1,l2,l4}$	◐		◐
SimBlock [AOK+19, BS19]	◐				◐				●	◐	●$_{k1}$	●$_{l1,l2,l4}$	◐		●
CIDDS [LNJ18]		◐								●				●	●
HIVE [hiv]				◐	◐	◐		◐	●	●	●$_{k1,k2}$	●$_{l1,l2,l3,l4}$			◐
Androulaki et al. [AKR+13]		◐							●	◐		●$_{l1,l4}$	●		
PeerSim [MJ09]														●	●

As a first simulator, let us mention Bitcoin-Simulator [GKW+16], which has been designed for educational purposes to help students to understand how the block generation rate (l_1) and block size (l_2) evolve over time. Although it is a well-designed pedagogical tool, it is quickly limited to carry out in-depth simulation analyzes. More research-oriented simulators have been proposed, such as Ethereum Hive [hiv] that have been proposed for emulating and evaluating Ethereum's smart contracts from a validation & execution time perspective (d_1-d_2). However, as revealed in [DNJ18], simulating a large number of nodes becomes difficult with limited computational resources. To overcome this limitation, a series of simulators including VIBES [SZJ17], eVIBES [DNJ18], eVIBES Plasma [evi] ones, as well as CIDDS [LNJ18], were proposed, but unfortunately fall short of fulfilling the promise, as reported by Lathif et al. [LNJ18]. One of the main reasons for this is that those simulators do not adequately model the transactions at the Node/Data layer (i.e., k_2), considering them as empty in the majority of the simulators. Such a consideration poses several issues when evaluating the blockchain system as a whole, knowing that transaction and block sizes may have non-negligible impacts on the overall system performance [LKG21]. To overcome this issue, a new range of simulators, including BlockSim [AvM19, FC19], SIMBA [FMF20], DLSF and

others (cf., Table 6.2), have considered the transaction/block structure, algorithms for wallet creation, message signing, and so forth, thus leading to more realistic performance evaluation results. Nonetheless, the network modeling is often too simplistic, not reflecting the real behavior of the P2P overlay network (i.e., possible communication delays, network congestion, packet losses, computation resource limitations, etc.), which, in our opinion, can have significant impact on the overall system performance evaluation process. A few simulators have been designed to consider the throughput and the network graph evolution, such as CIDDS, PeerSim. However, these two simulators neglect many of the upper layers and metrics, as highlighted in Table 6.2.

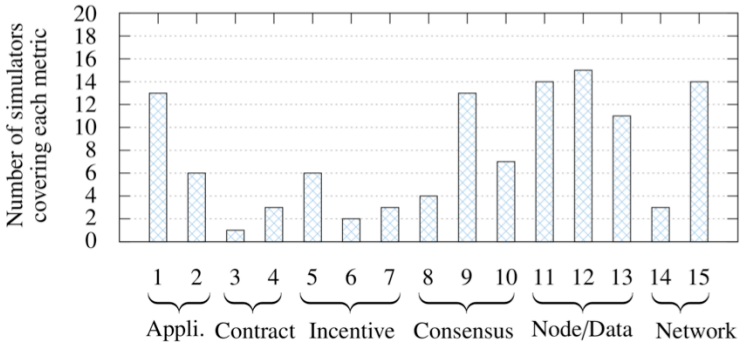


Figure 6.2: Overview of the extent to which state-of-the-art simulators – *the* 17 *reported in Table 6.2* – cover the six-layer model introduced in Figure 6.1.

The drawbacks discussed in this section prove that designing a modular simulator that allows for testing/evaluating different blockchains, with different consensus protocols, different contract and transaction/block specifications, different incentive schemes, along with network infrastructures, turns to be a challenging task. Figure 6.2 provides an overview of the extent to which the reported blockchain simulators cover the different layers of the six-layer model previously introduced. It can be first observed that they primarily often neglect the Contract and Incentive layers. Second, key aspects of a blockchain systems at the Network and Data/Node layers, namely the consideration of the real P2P overlay network protocol behavior (n) and transaction/block structure (k_1), result in non-optimal performance evaluation results (non-optimal compared to the reality). A new hybrid emulator/simulator tool called “BlockPerf" is proposed in this paper to overcome this limitation. BlockPerf is hybrid in the sense that it emulates the network layer to correctly address the network layer while simulating the upper layers based on statistical data modeling approaches, as presented in the next section.

6.3 BlockPerf simulator design

This section describes how BlockPerf extends BlockSim, whose main extensions are summarized in Figure 6.3. These layer extensions and/or adaptations are respectively discussed through sections 6.3.1 to 6.3.5. Note that, at this stage, BlockPerf does not address/model the “Smart Contract" layer and is part of future research work.

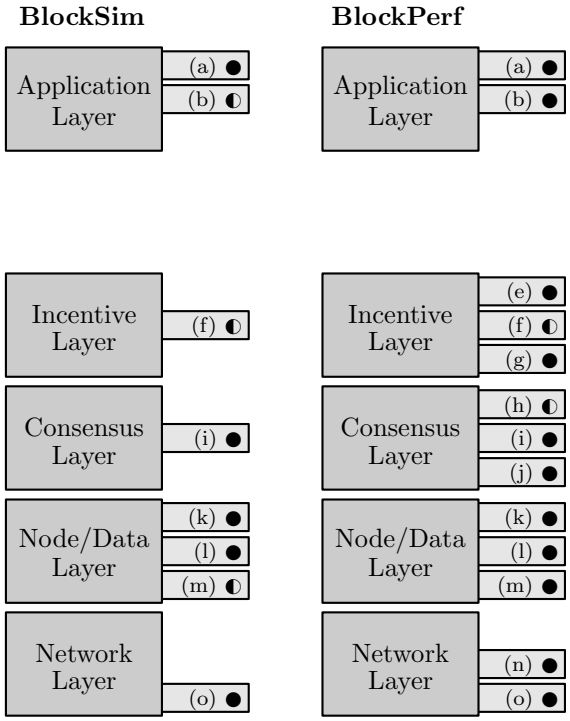


Figure 6.3: Illustration of how BlockPerf extends BlockSim - ● symbol indicates that the simulator makes available the performance metric without any code modification and/or post-processing, while ◐ symbol indicates that such a modification and/or post-processing is needed to obtained the desired performance metric

6.3.1 Application layer

At this layer, BlockPerf consists of a configuration module responsible for instantiating the run's parameters similar to the one used in BlockSim. BlockSim takes as input a list of parameters, including the size of blocks, statistical models for transaction validation, block validation, and node labels. Similarly, BlockPerf takes as input a `JSON` file that extends the parameters from BlockSim to include fork occurrences, the block size parameter, IP address of each node, including

the 'main" one (corresponding to the `BitcoinNode` class, as summarized in Table 6.3) that tracks all the results using the `CliStats`, `CpuTimeSnapshoot` and `MemorySnapshot` classes. Optionally, the input file also considers the location to store all the logs and data for each node. Note that the `BitcoinNode` is responsible for tracking the execution time (a) of the overall run and collecting the metrics regarding the overall computational resource usage (b) of the nodes.

Table 6.3: Technical description of the major classes within BlockPerf, along with the layers they cover.

Class Name	Description	Application	Incentive	Consensus	Node/Data	Network
CliStats	Node state checking and logging (incl., parameters like operation time, computation)	✓		✓	✓	
CpuTimeSnapshot	Computation process and database tracking	✓			✓	✓
MemorySnapshot	Output file generation for the memory used by a given node	✓				
Proxy	Authentication management for network components and wallet information	✓				
TxChain	Chain operation handling (address generation, private-public keys, unspent Tx, etc.)		✓	✓	✓	
PublicBitcoinNode	Network stat monitoring (network graph, latency...)				✓	✓
BTCNode	Monitoring of transactions/block evolution, handshaking, etc.		✓		✓	✓
Node	Node operation handling (incl., creation of transaction, Queues)				✓	✓
TickEvent	Transactions/block timestamping		✓		✓	
TransactionQueue	Queue transaction creation (i.e., transactions to be generated by a given node)				✓	
BlockHeader	Header tracking for the chain that a node follows			✓		✓
Consensus	Consensus protocol model instantiation		✓	✓		
BitcoinNode	Connection handling and in charge of pushing model parameters to the network					✓
ZoneConfig	Node parameter consideration for the emulation layer (incl., IP addresses)					✓
Event	Event tick creation based on the models					
Runner	*main* class					

6.3.2 Incentive Layer

This layer is responsible for instantiating the models for rewarding participants, which vary from one blockchain to another. While BlockSim does not simulate the incentive layer[2], BlockPerf models two types of rewards: one for the generation of the valid blocks (known as *block reward*) and the second for the inclusion of a particular transaction to the block (known as *transaction fee*). These reward operations are implemented via the `TxChain`, `BTCNode` and `TickEvent` classes (*cf.*, Table 6.3. As of today, the *block reward* is represented as a fixed amount of cryptocurrency that can be configured for the run, while the *transaction fee* is dependent on the size of the transaction, as formalized as in Eq 6.1, where Tx_i refers to the i^{th} transaction and $f\left(\text{size}(\text{Tx}_i)\right)$ can be configured using different distributions laws (e.g., defined as a fixed fee, or following a Weibull, Log-normal, or Gamma distribution). All node wallets are continuously updated throughout

[2]Even though the authors in [AvM19] state that the incentive layer is partly covered, nothing from the source code provides clear evidence of this.

the simulation run.

$$\mathrm{Tx}_i^{\mathrm{fee}} = f\left(\mathrm{size}(\mathrm{Tx}_i)\right) \tag{6.1}$$

The amount of new cryptocurrencies generated over time depends on the consensus layer and the overall computational resources of the nodes in the network. BlockPerf continuously monitors such resources and allocates the newly generated cryptocurrencies to the right (mining) nodes.

6.3.3 Consensus Layer

In BlockPerf, the consensus protocol is implemented as a model that represents the consensus process and its operations. The consensus algorithm is coded within the `Consensus` class (see Table 6.3), which is in charge of selecting the miner/validator that builds the next block. The method of selecting the node varies from one blockchain to another, although in BlockPerf, as a first step, the Proof-Of-Work (PoW) algorithm has been implemented. As opposed to BlockSim that simulates the behavior of the consensus algorithm (i.e., the block validation and a random selection of miner), BlockPerf uses an extended approach where each mining node, similar to PoW algorithm, selects a random number, the input for all the non-confirmed transactions from its queue within the limit of block size (further discussed in Section 6.3.4), as well as the reference of the previous known block, which allows for being closer to a real PoW process.This information is then combined and hashed recursively until a result is obtained. The mining node selects the transactions from the queue (sometimes called mempool), check whether they meet the balance requirements, verify whether the sender signatures match and that the sender's wallet has a sufficient amount of cryptocurrencies. In BlockPerf, the `BitcoinNode` class keeps track of the fork occurrences within the network.

6.3.4 Node/Data Layer

Any node wishing to generate a transaction has to follow a set of operations, namely: (i) select a wallet address (from *wallet_list*) as the recipient; (ii) select a random value for the transaction; (iii) generate the transaction by signing it using its private wallet key (those operations being handled via the `TickEvent`, `Transaction Queue` and `TxChain` classes, as reported in Table 6.3). The set of transactions to be created per unit of time t, which is denoted by $\mathcal{X}(t) = \{\mathrm{Tx}_1, \ldots, \mathrm{Tx}_k\} \mid k \in \mathbb{N}$, can be configured using different distributions laws (e.g., a fixed amount of transactions per unit of time, or following a Weibull-like distribution). Every new transaction created by a node calls the propagation function from the network layer to transmit this new transaction to its neighbors, who then append it to their pool. Upon reception of a transaction (Tx_i), several operations are undertaken by the recipient node, as summarized in the flow chart

Figure 6.4 (operations that have been added compared with BlockSim being highlighted in green). Unlike BlockSim, BlockPerf models transactions as they exist in the real system (i.e., each transaction can be traced back and follow a similar process within the system).

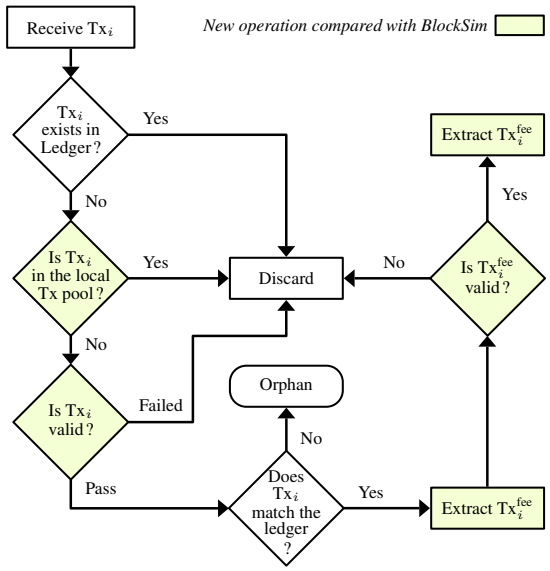


Figure 6.4: Flowchart of transaction operation sequence upon reception of Tx_i in BlockPerf

As a second stage, the simulator has to handle the inclusion and/or removal of transactions – *from its transaction pool* – into blocks. Although the consensus layer governs operations towards deciding which block to be accepted by everyone within the network, the Data Layer is concerned with the reception of the block, being in charge of removing all the transactions which are enclosed in the last received block from its transaction pool, and finally attaching the new block to its last known state of the chain (handled by the `Node` class). These operations follow the sequence of operations given in the flow chart of Figure 6.5 (new operations compared with BlockSim being highlighted in green, while BlockSim's operations that have been modified are highlighted in orange).

The linked blocks form a chain, but nodes may have a different state of the chain (or ledger) due to network communication delays (this is termed forks). As opposed to many existing simulators, including BlockSim, each node in BlockPerf has its local ledger implemented, which means that if a node discovers that the state of the chain is different from its own, it sends a request to neighboring nodes for getting the full state of the chain. This allows being closer to reality, which should result in more realistic simulation performance. This improvement,

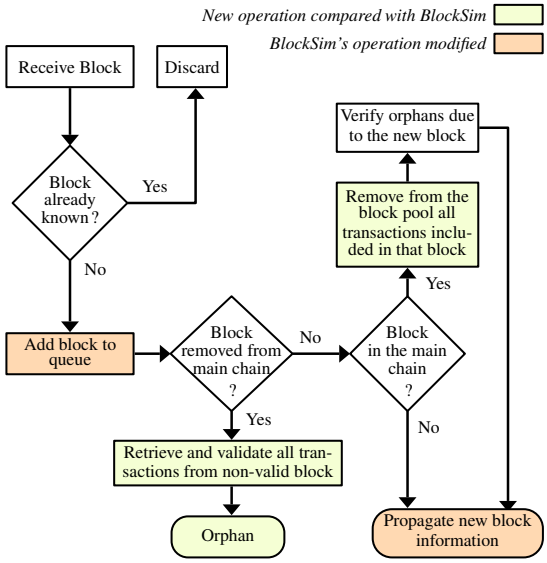


Figure 6.5: Flowchart of transaction inclusion/removal from a block in BlockPerf

compared with BlockSim, is highlighted in the flow chart given in Figure 6.6 (*cf.* green boxes/steps).

6.3.5 Network Layer

The network layer of BlockPerf differs significantly from BlockSim, and many other simulators reported in Table 6.3, since, in BlockPerf, the blockchain network is emulated over distributed (physical) nodes. The benefit of doing so is that the simulation should lead to results closer to reality, as BlockPerf implements a P2P protocol similar to the ones used by real blockchain systems using advertisement messages (e.g., `GETADDR` and `ADDR`) to discover neighboring nodes. BlockPerf also integrates:

- *a wallet management module:* in charge of generating wallet addresses to uniquely identify all nodes. This module also integrates the protocol to broadcast wallet-related information in the network (via `WALLADD` messages) to make all nodes aware of existing addresses;
- *a reputation score module:* in charge of assigning negative scores from the connections where malformed messages were received (e.g., if the transaction correctness fails, nodes keep receiving outdated block information). This module is fulfilled by the `PublicBitcoinNode` class, which is responsible for keeping a log of the neighboring nodes and associated scores. Such logs allow BlockPerf to keep track of the network graph evolution (n) over the simulation run.

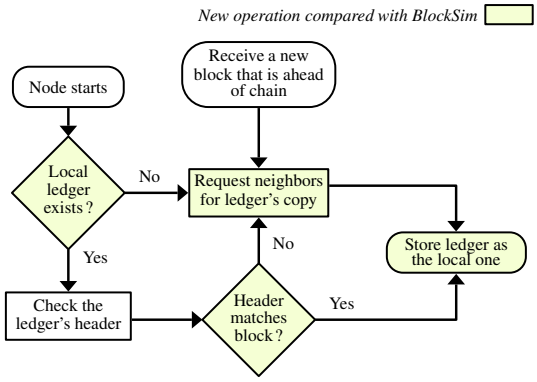


Figure 6.6: Flowchart of local ledger in BlockPerf.

6.3.6 BlockPerf Implementation

Table 6.3 provides an overview of the important programming classes underpinning BlockPerf, whose implementation is combination of C/C++ programming languages for the network layer, and Python for the upper layers. The source code of the BlockPerf simulator is available on GitHub[3].

6.4 Validation and Evaluation

This section aims to compare the extent to which BlockPerf provides realistic results, but also to which it outperforms BlockSim. Figure 6.7 provides an overview of the experimental process that has been conducted in this respect, which consists of five main stages denoted by ① to ⑤ in Figure 6.7. First (①), a real-life blockchain network (`bitcoind`) consisting of 63 nodes spread over the world has been implemented, which corresponds to the benchmarking Testbed in this study whose metrics (a) to (o) have been collected/measured whenever possible (see ②). Based on those metrics, several parameters are specified as input parameters of BlockPerf and BlockSim (see ③ and ④), as will be presented in section 6.4.1. BlockPerf and BlockSim are then compared against the results obtained for the (benchmarking) Testbed in section 6.4.3 (see ⑤). A discussion about our experiments is finally given in section 6.4.3.

6.4.1 Benchmarking Testbed

The (benchmarking) Testbed is made of a modified version of the reference bitcoin implementation, namely `bitcoind`[4]. The evaluation is performed during 11 days, whose blockchain network consists of 23 full nodes and 40 light nodes

[3]BlockPerf repo: https://github.com/Deadlyelder/BlockPerf

[4]A slight difference can be noted, as our testbed is a private network where all nodes are controlled and maintained throughout the experiment.

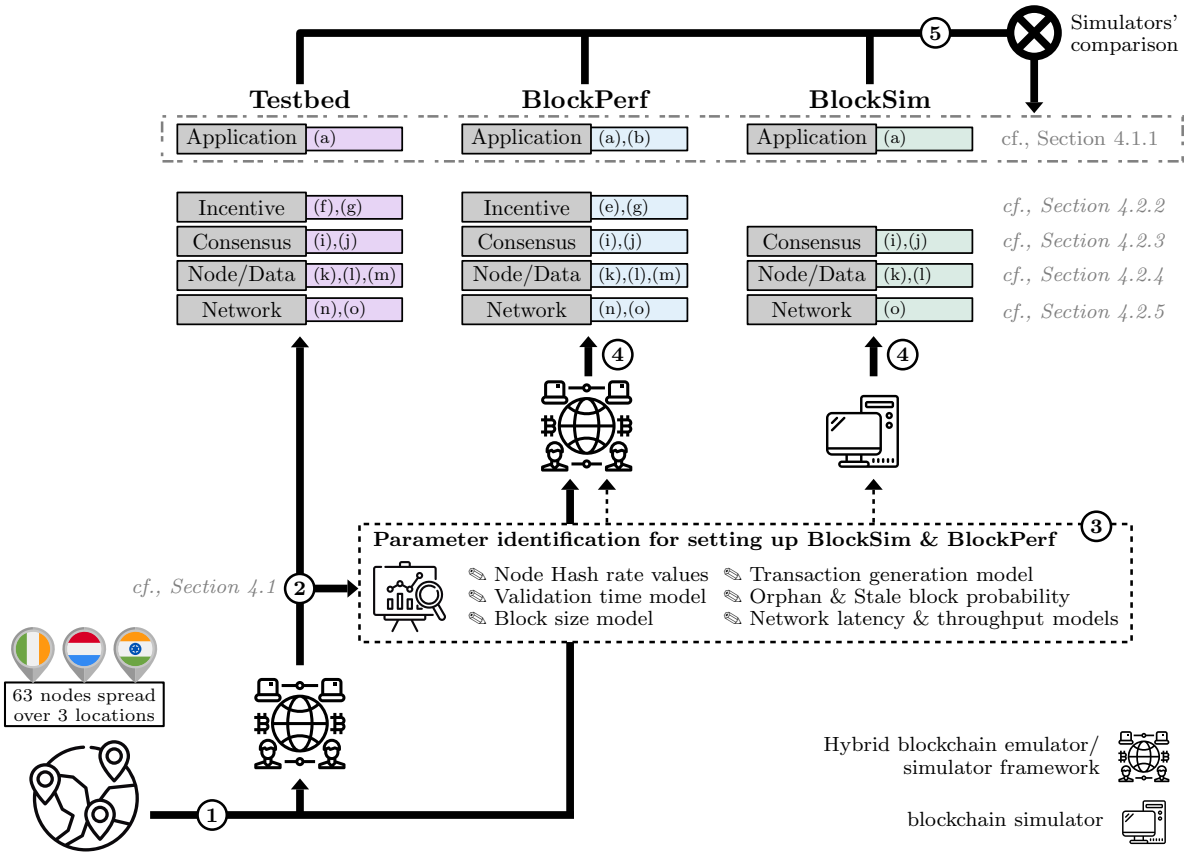


Figure 6.7: Experimental process set up and conducted in this paper to compare BlockSim and BlockPerf simulators

spread over three locations, namely Ireland, Luxembourg and India. During the course of the experiment, transactions are randomly created. Based on this experimental configuration setting, metrics (a) to (o) are measured, whenever possible, and then reused for two purposes:

1. to serve as benchmarking metrics to evaluate the extent to which BlockPerf and BlockSim deviate from the reality (see ⑤ in Figure 6.7);

2. to identify, from the testbed run, several parameters that need to be configured as inputs of BlockPerf and BlockSim such as node hash rate values, block size distribution model, etc. (see ③-④). To this end, a similar process as the one defined in [FC19] has been applied to extrapolate the probability distribution model for each parameter.

Table 6.4 summarizes all the metric values obtained from the Testbed, and, for some of them refer to specific figures/graphs. For example, looking at metric

Table 6.4: Overview of the results

	Metrics	Evolution (cf.,)	Testbed	BlockPerf	BlockSim
Application	(a) Execution time (in hours)	-	264h	8h	0.5h
	(b) Computation usage (Average CPU)	-	80%	90%	-
	(Average memory)	-	40%	86%	-
	(Average storage)	-	2.6GB	2.9 GB	-
Incentive	(e) Average reward per day	Figure 6.8	-	0.043991	-
	(f) Fee evolution	-	*by default*	0.00001	-
	(g) Currency generat. rate per time unit	Figure 6.9	720 BTC	384 BTC	-
Consensus	(i_1) Total number of forks	-	2	5	2
	(i_2) Total number of blocks discarded	Figure 6.10	4	13	5
	(j) Min-Max consensus computation effort	-	10-45 MH/s	9-42 MH/s	1-2 MH/s
Node/Data	(k_1) Average Tx/day	Figure 6.11	7341	4504	14184
	(l_1) Total number of validated blocks	Figure 6.12(a)	1979	1118	4132
	(l_2) Average time to validate blocks	Figure 6.12(b)	10.7min	10.9min	10.4min
	(l_3) Average block size	Figure 6.12(c)	1.6 MB	1.1 MB	2 MB
	(l_4) Average number of Tx/block	Figure 6.12(d)	41	44	38
	(m) Chain's length (total number of blocks)	Figure 6.13	1980	1119	4133
Network	(n) Network graph evolution (Cluster Coefficient)	-	0.473217	0.469478	1
	(Mean Geodesic Distance)	-	2.08631	2.11437	1
	(Diameter)	-	4	4	1
	(o) Average throughput (Tx/s)	Figure 6.17	1.5	1.6	2

k1, Table 6.4 reports the average number of transactions/day, while Figure 6.11 reports the exact number transactions over the 11 days of the experiment).

6.4.2 Experiment with BlockPerf and BlockSim & comparison

In this section, the performance of BlockPerf and BlockSim is compared against the benchmarking testbed results. BlockSim experiments have been conducted on a single machine with a 2.40 GHz Intel Xeon E5 CPU and 4GB of RAM. In contrast, BlockPerf has been deployed over distributed nodes – *located in Ireland, Luxembourg and India* – with the same computational resource (i.e., 2.40 GHz Intel Xeon E5 CPU and 4GB of RAM). Sections 6.4.2 to 6.4.2 discuss the comparison analyses regarding the different layers, as emphasized in Figure 6.7 (see ⑤). Note that for this comparative analysis, only the metrics that are covered and for which the results are available (i.e., metrics with ● in Figure 6.3 and listed in Figure 6.7) are considered.

Application Layer

From an *Execution time (a)* perspective, BlockSim and BlockPerf allow for simulating the experiment at a faster pace compared to the testbed, the former being 528 times faster while the later is 33 times faster. This substantial difference is explained by the fact that BlockSim does not rely on a real-life network layer. In terms of *Computational resource usage (b)*, the average CPU, memory and storage metrics have been reported in Table 6.4, although it cannot be concluded that one simulator is better than another for those metrics.

Incentive Layer

Unlike BlockSim and the Testbed, BlockPerf stores log about the *Reward evolution (e)*, as they are generated within the run. Figure 6.8 provides insight into the evolution of the reward over the 11 days of simulation. On average, the reward is 0.043991 BTC per day, equally distributed to the miners with a standard deviation of 0.0002 throughout the run. Although no comparison for this metric could be made with the Testbed and BlockSim, we nonetheless report those results as they could serve as benchmarks for other researchers.

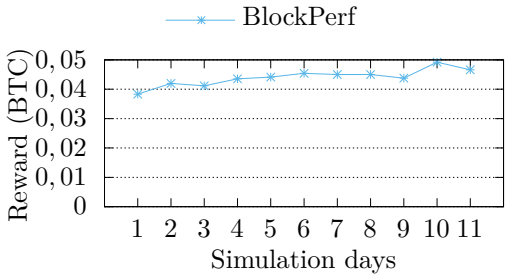


Figure 6.8: Reward evolution over days (e)

The *Currency generation rate (g)* can be compared with the Testbed, but not with BlockSim (metric not available). Figure 6.9 shows the evolution of the currency generated over the course of the simulation, 384 BTC being generated on average by BlockPerf, against 720 BTC generated by the Testbed. Despite this difference, what is interesting to note is that that the evolution in BlockPerf and Testbed follows a similar trend.

Consensus Layer

During the simulation, a number of *Fork resolutions* (i_1) are handled by the Testbed (2 in total, one at day 4 and one at day 5), by BlockSim (2 in total, one at day 4 and one at day 5), and by BlockPerf (4 in total, three at day 5 and two at day 6). These forks led to the appearance of stale blocks (i_2), whose occurrences over the 11 days of the experiment have been reported in Figure 6.10. Two observations can be made: (i) stale blocks mostly appear in the same timeframe (between day 4 and 6); (ii) BlockPerf provides more realistic results than BlockSim, as the number of stale blocks in BlockPerf (4 in total) is closer to the Testbed (6) than BlockSim (13).

Regarding now the *Consensus computation effort* (j), BlockPerf also provides more realistic results, ranging between 9-42 MH/s, while the Testbed ranges between 10-45 MH/s and BlockSim between 1-2 MH/s. The reason for such a difference is that, in BlockSim, even when specifying as input parameters the min and max hash rate values (i.e., 10-45 MH/s identified through the benchmarking phase,

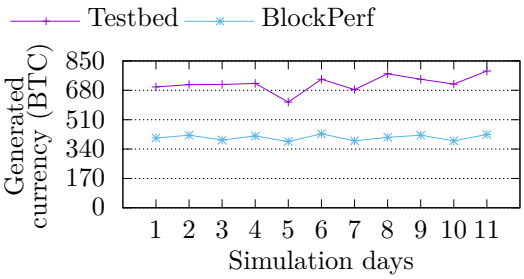


Figure 6.9: Currency generated over days (g)

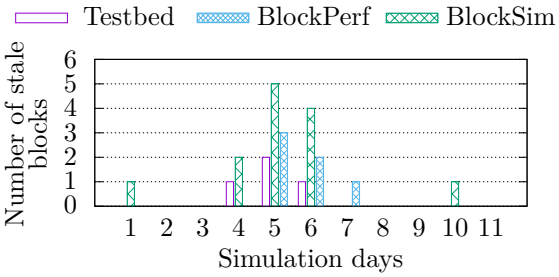


Figure 6.10: Stale blocks per days (i_2)

cf. section 6.4.1), it seems that BlockSim does not consider that parameters in the simulation run, always applying 1-2 MH/s as hash rate range.

Node/Data Layer

As the first metric of that layer, let us look at the *Transaction evolution* (k_1) in Figure 6.11, which shows the number of transactions generated daily by the Testbed and the two simulators. It can be observed that BlockPerf closely follows the Testbed's transaction evolution, thus providing more realistic results than BlockSim. Indeed, BlockPerf and BlockSim respectively generate 4504 and 14184 Tx/day on average, while the Testbed generates 7341 Tx/day. For a more accurate view on the extent that BlockPerf leads towards realistic results, we show the relative absolute errors of BlockPerf and BlockSim with respect to the results obtained from the testbed is represented in the form of a boxplot in Figure 6.11. For example, the max value of the BS (BlockSim)-related boxplot (i.e., ≈ 12000) indicates that, over the 11 days of experiments, the maximal difference in terms of number of transactions generated by the Testbed and BlockSim is 12000, which – *if we look at the transaction evolution in Figure 6.11* – corresponds to day 5. Looking at the error median values, it can be noted that the difference/error is quite significant in BlockSim, being approximately twofold higher compared to BlockPerf. In addition of k_1, let us also note that BlockPerf follows the *real-life transaction specification* (k_2), while BlockSim adopts a more simplistic model, which may ex-

plain part of the difference in results of the transaction evolution above-discussed (k_1).

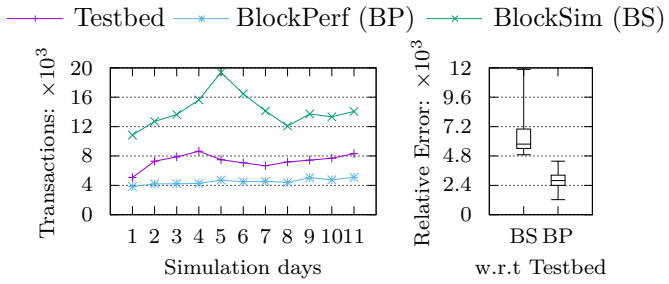
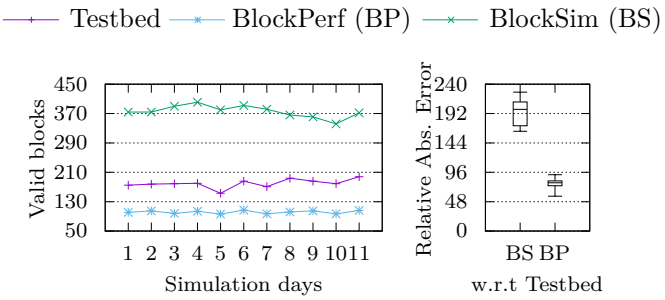


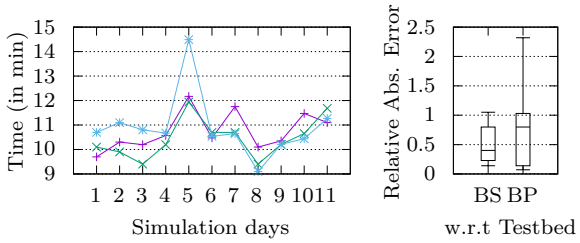
Figure 6.11: Transaction evolution (k_1) & relative absolute error with respect to (w.r.t) the Testbed

Let us now discuss about the *Block evolution* (l), which consists of four sub-metrics (l_1 to l_4). Figure 6.12(a) shows the evolution of the number of valid blocks (l_1) over the 11 days of experiment. It can be observed that BlockPerf produces fewer blocks than the testbed, which is likely due to the unpredictable nature of the network connections and the occurrence of forks. On the other hand, BlockSim produces a number of blocks (≈400/day) that is twice higher than the Testbed (≈200/day). Overall, as evidenced through the boxplot in Figure 6.12(a), BlockPerf provides more realistic results than BlockSim, whether in terms of min, quartiles and max values. Looking now at the second sub-metric (l_2: block validation times), both BlockPerf and BlockSim provide similar results to the testbed, as shown in Figure 6.12(b). Indeed, the three curves followed the same trend and the relative absolute error was ≤ 1 min (i.e., one-tenth of the total time required by the Testbed). Figure 6.12(c) then provides insight into the evolution throughout the run of the block validation times (l_3), along with the difference – *relative absolute error to be precise* – between BlockPerf/BlockSim and the Testbed. It is interesting to note that, when looking at the boxplot, BlockSim provides closer results with the Testbed than BlockPerf; however, when looking at the day-by-day block validation times, BlockPerf evolves in a similar way to the Testbed, while BlockSim does not. Finally, regarding l_4 (i.e., the average number of transactions mined within blocks per day), the two simulators provide similar results to the Testbed, following a similar trend over the course of the run and having similar differences/errors compared to the Testbed (the min and max errors being the same for BlockPerf and BlockSim).

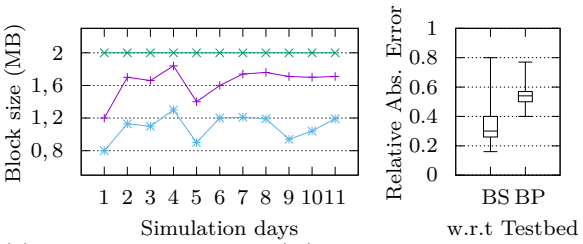
The last metric of the Node/Data layer refers to the length of the chain (m), whose evolution over the 11 days of the experiment for the Testbed and the two simulators are plotted in Figure 6.13. It can be observed that BlockPerf provides more realistic results than BlockSim, which becomes increasingly significant over



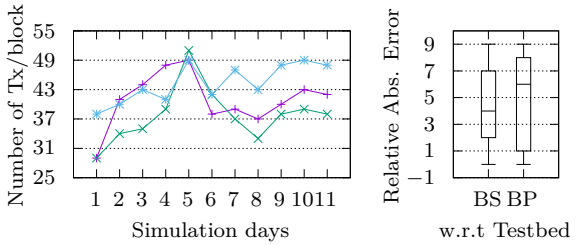
(a) Number of valid blocks (l_1) & relative absolute error w.r.t Testbed



(b) Inter-block average time (l_2) & relative abs. error



(c) Average block sizes (l_3) & relative absolute error



(d) Transactions per block (l_4) & relative absolute error

Figure 6.12: Block evolution (l_1-l_2)

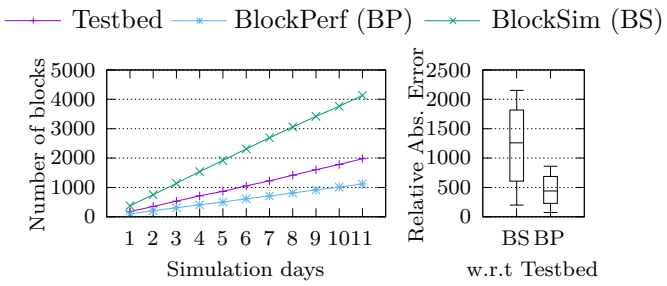


Figure 6.13: Chain's length (m)

time. Indeed, the longer the simulation run, the higher the difference between BlockSim and the Testbed. This is also confirmed by the boxplot given in Figure 6.13, the minimal error values of the two simulators being approximately the same, while the quartiles, median and max values become increasingly higher for BlockSim.

Network Layer

Two distinct metrics are analyzed at this layer, namely how the network graph evolves over time (n), and what throughput performance is possible with the implemented blockchain and scenario (o).

Regarding the first metric (n), Figures 6.14, 6.15 and 6.16 provide the network graph evolution over two consecutive days (days 1 and 2) respectively regarding the Testbed, BlockPerf and BlockSim. Pink nodes correspond to light nodes and yellow ones to full ones, each node being denoted by the country's initial (L: Luxembourg, I: Ireland; In: India) and the node's numbering. A twofold observation can be drawn from those graphs. First, unlike BlockPerf and the Testbed, BlockSim assumes that all nodes are interconnected in a fully connected mesh throughout the experimental run, adding that it does not distinguish lightweight and full blockchain nodes. A second observation is that the P2P logical network infrastructure evolves quite substantially from one day to another, whether regarding the Testbed or BlockPerf, which are due, among other things, to eventual connection losses, ongoing traffic,.... The Diameter[5], mean geodesic distance[6], and clustering coefficient[7] reported in Table 6.4 show that the graphs evolve in a similar manner

[5]Diameter: $max\left(dist\left(n_i, n_j\right)\right)\ \forall\ n_i, n_j \in \mathcal{N}$, with $\mathcal{N}$ the set of nodes (full and lightweight) from the network graph.

[6]Mean geodesic distance: $mean\left(dist\left(n_i, n_j\right)\right)\ \forall\ n_i, n_j \in \mathcal{N}$, with $\mathcal{N} = \{n_1, \ldots, n_N\}$ the set of nodes (full and lightweight) from the network graph.

[7]Clustering coefficient of entire graph: $C = \frac{1}{N}\sum_{i=1}^{N} C_i$, where C_i refers to the clustering coefficient of a node n_i that is calculated as follows: $C_i = \frac{2.L_i}{k_i(k_i-1)}$, k_i referring to the degree of node n_i and L_i to the number of edges between the k_i neighbors of n_i.

for BlockPerf and BlockSim, which is not the case for BlockSim.

Throughput is a key metric that provides a good performance indicator of a given blockchain system. It is nonetheless tricky to analytically formalize it, as it depends on multiple processes across the different layers. Indeed Application (for generation), Node/Data (for accessing rewards), Consensus (mining), Incentive (for rewards), and Network layers (for final throughput), all combined, influence the throughput of a given blockchain system. Figure 6.17 shows the evolution of the average throughput for the considered scenario, along with the boxplot showing the relative absolute errors of BlockSim and BlockPerf when compared to the Testbed. The results show that throughput ranges between 1 and 2.5 Tx/s for both the Testbed and the two simulators (BlockPerf having slightly closer results to the Testbed than BlockSim).

6.4.3 Discussion

The previous section has shown that BlockPerf outperforms BlockSim for most of the metrics (a)-(o), or provide more realistic results to be more precise. We firmly believe that this is mainly due to the assumption made by BlockSim (i.e., non-consideration of the network-level change within the system). Furthermore, our approach presents several advantages. First, it covers all the layers (except the contract one) and allows to obtain a higher number of performance metrics. Also, by emulating the network layer, our approach replicates a more realistic behavior, thus leading to more realistic results than BlockSim. Finally, even if it is not yet the case, our approach is aimed at making the plugging of any blockchain technology easier, as our code is structured following the six-layer model described in Figure 6.3.

One may wonder whether the number of nodes used in our experiments is not too small. This parameter is not critical in this study as the objective is not about showing how efficient a given blockchain is (e.g., from a scalability standpoint), but rather on showing that the proposed simulator provides more realistic evaluation results than an existing one (BlockSim in this case). As a consequence, we limited the experiment to be large enough to capture a realistic scenario, and short enough to be manageable (in terms of costs).

One may also wonder whether the size of the datasets is large enough to draw relevant conclusions about the comparison analysis. In this regard, let us note that all the graphs given in Figures 6.8 to 6.17 only report the average values of all the measurements obtained on a daily basis for the corresponding metrics. Just to give an indication about the number of measurements that have been collected every day, around 10000 measurements/day were collected regarding metrics (e) and (k_1), 200 measurements/day regarding (l_1)-(l_3), and about 50 regarding l_4. Given this amount of measurements, we confidently state that the conclusions drawn from our comparison analysis are relevant (statistically speaking).

6.5 Conclusion, Limitations & Perspectives

6.5.1 Conclusion

Today, a large variety of blockchain technologies is expanding in order to fulfill technical and non-technical needs and requirements. Within this context, determining and, most importantly, evaluating the characteristics/performance of a given blockchain platform is crucial for system designers before deploying it. In this respect, several blockchain simulators have been proposed in the literature over the past few years. Still, they are often limited in several respects (lack of extensibility, failure in covering all aspects/metrics underpinning a blockchain system, etc.). In this paper, the six-layer model introduced by [DWC+17] (Application, Contract, Incentive, Consensus, Node/Data, Network) is considered, against which 15 performance metrics have been mapped.

To overcome the limitations of state-of-the-art blockchain simulators, a new one called BlockPerf is proposed in this paper, an extension to the existing BlockSim simulator. BlockPerf tries to cover as much as possible the layers mentioned above along with its metrics. A comparative analysis with BlockSim is carried, which has demonstrated that BlockPerf provides more realistic results than BlockSim (e.g., at the Node/Data and Network layers) respectively improved by 39% and 55% in average. However, several limitations remain to be addressed in the future, as discussed in the next section. The comparison analysis conducted in this paper is built upon a benchmarking bitcoin scenario.

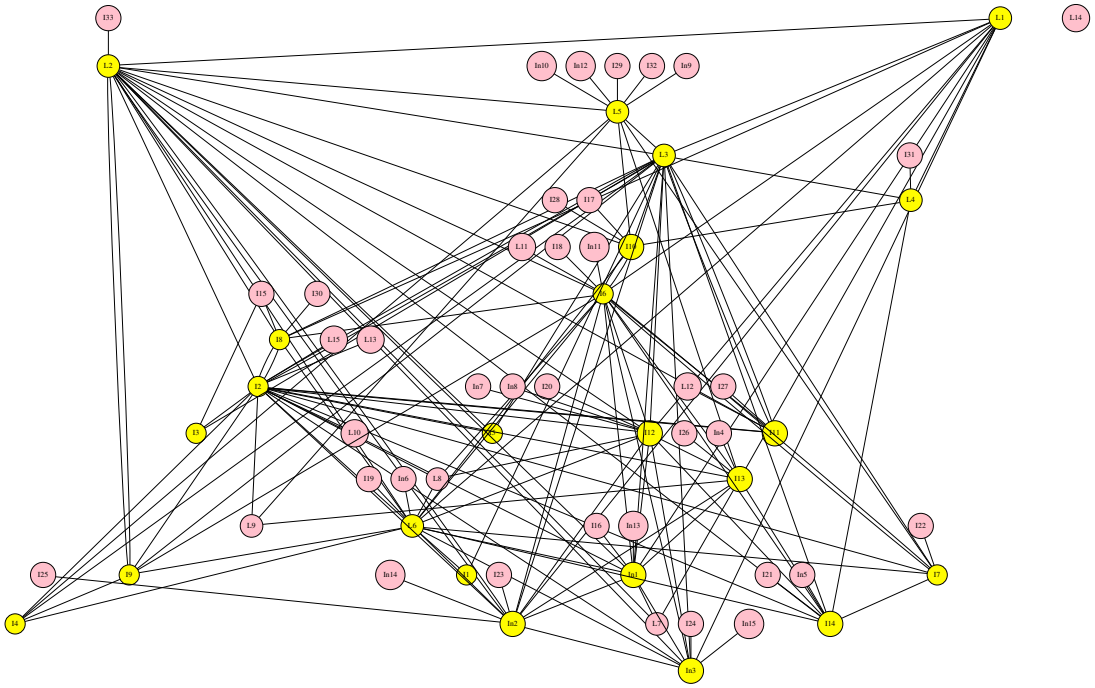
6.5.2 Limitations & Perspectives

Several limitations to BlockPerf are to be addressed for it to be a more exhaustive simulator. First, the deployment cost of deploying nodes within BlockPerf on different geographical locations is not a simple and straightforward task, as it requires careful planning and deployment of nodes, connecting them with the main interface within BlockPerf. However, such a layer is, in our opinion, crucial to obtain simulation results closer to reality.

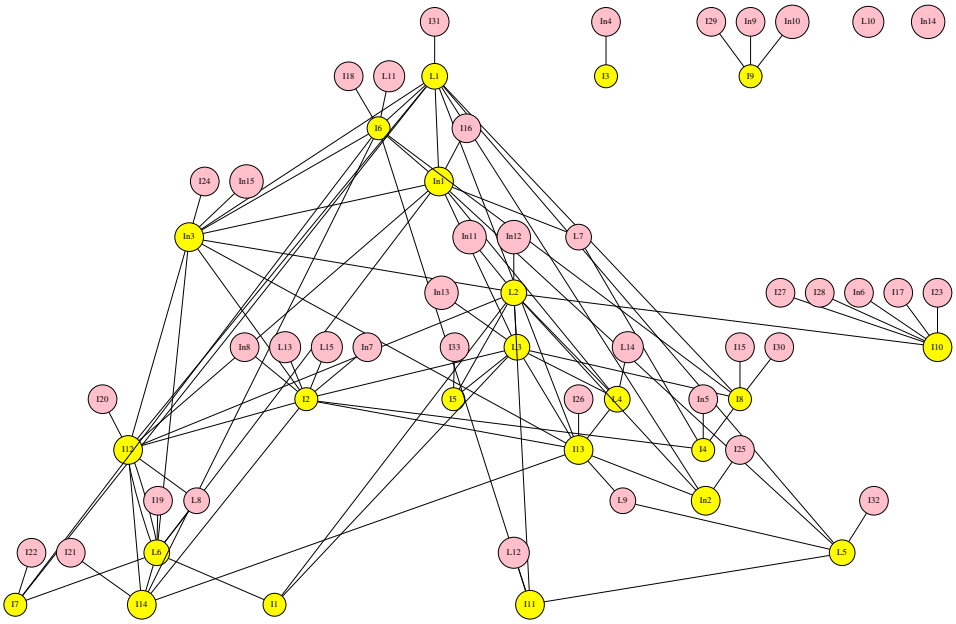
Second, BlockPerf, in its current form, only supports Bitcoin-related experiments, and some efforts still remain to be done to allow for simulating different blockchain platforms. For example, a blockchain-based system such as the IOTA, which uses graph-based transaction chains, would require more extensive tuning of BlockPerf to replicate the effects of its real-world deployment.

Third, the *Contract layer* still remains to be developed within BlockPerf. The ability of deploying contracts has been a key element and argument within all the blockchain technologies that emerged over the last years, and being able to analyse them within the simulation environment would likely provide further insights on its usage. However, covering/integrating such a layer in a simulator is particu-

larly challenging because each smart contract execution happens within a virtual computing environment (e.g., EVM in Ethereum), which is called by every node execute the instructions contained by that contract. This environment itself relies on other layers for execution, including the execution of certain transactions or changing the state of nodes, which makes it difficult to obtain realistic execution effects on the overall system, as discussed in [Xu21]. Given this complexity, we believe that the adoption of a hybrid approach that uses the virtual computing environment as an emulation layer – *in a similar manner as done in BlockPerf with the network layer* – is an efficient way to cover/integrate the smart contract layer in a simulator.

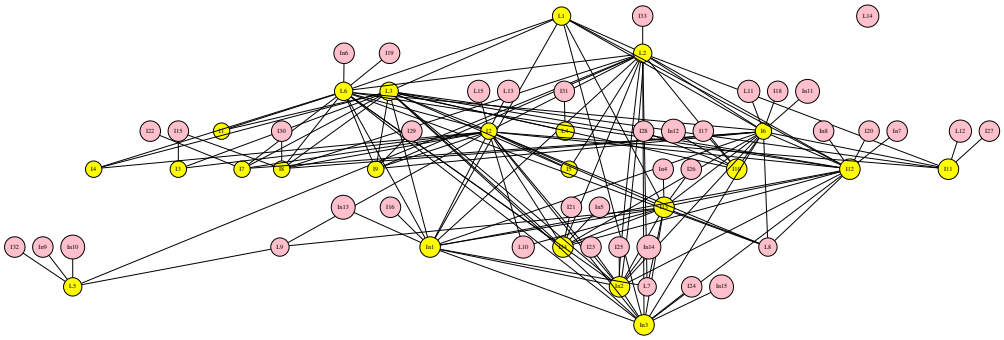


(a) P2P network topology underlying the Testbed at day 1 of the experiment

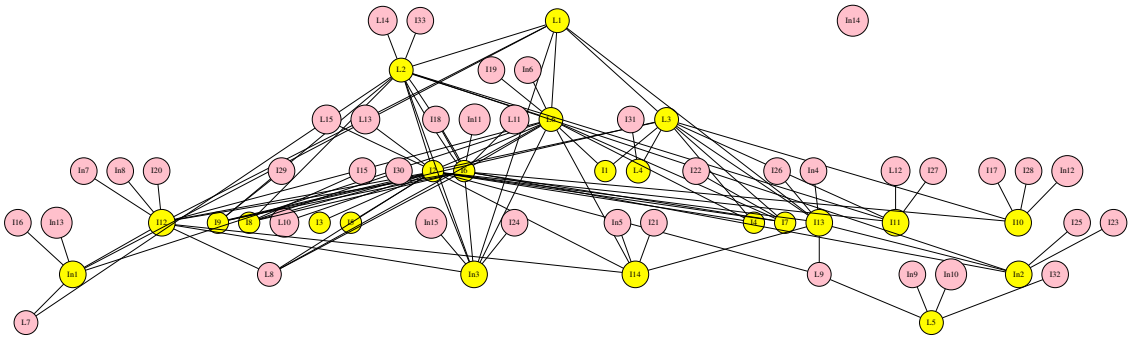


(b) P2P network topology underlying the Testbed at day 2 of the experiment

Figure 6.14: Testbed's network graph evolution over two days of experiment



(a) P2P network topology underlying BlockPerf at day 1 of the experiment



(b) P2P network topology underlying BlockPerf at day 2 of the experiment

Figure 6.15: BlockPerf's network graph evolution over two days of experiment

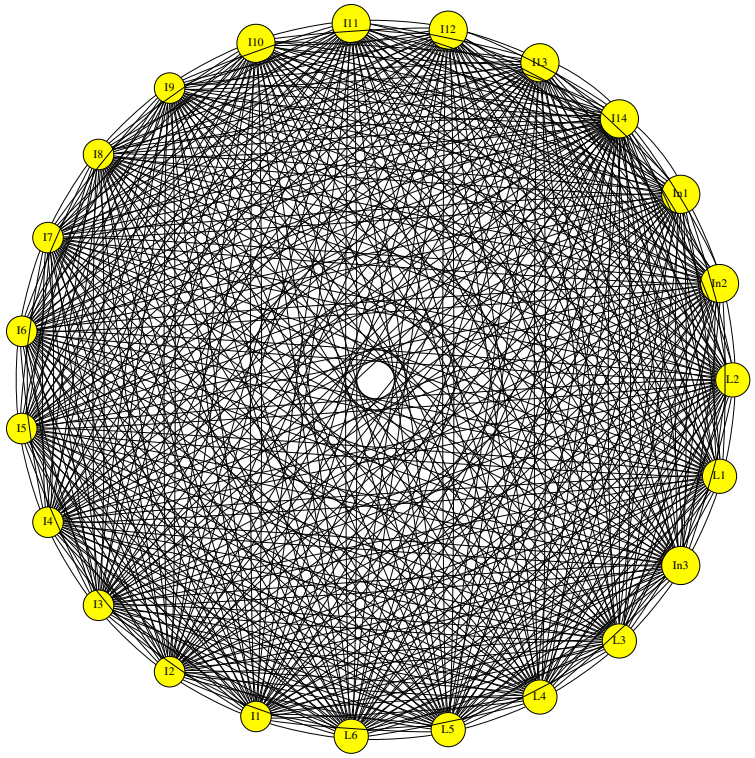


Figure 6.16: BlockSim's network graph remaining unchanged throughout the simulation run

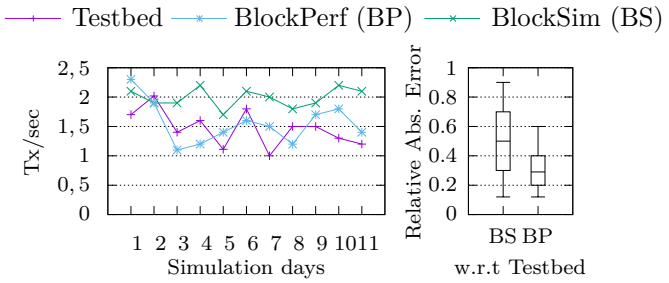


Figure 6.17: Throughput (o) & relative absolute error w.r.t Testbed

7

Conclusion & Perspectives

7.1 Conclusion

This dissertation is composed of three parts and aims at investigating the transition towards Industry 4.0 in *brownfield* manufacturing facilities.

The first part focuses on the foundation of this transition, which is a secure data collection architecture, allowing for the access to the production data from the enterprise network. More precisely, the first part addresses the security aspect of OPC-UA, the *de facto* standard for unifying industrial communications. The unification of the communication requires the interconnection of the OT network and the IT network. However, in *brownfield* environments, the OT network is composed of legacy vulnerable equipment that could compromise the IT network if the vulnerabilities are not identified and mitigating actions are not considered. In our study, we identify based on the standard specifications, the threats and mitigating actions that may occur/be applied when using OPC-UA in an Industry 4.0 environment and we highlight the impact of two different attacks on an OPC-UA application implemented on a real testbed. Overall, our experiments show that OPC-UA as middleware, implements satisfactory security mechanisms, but can not fill the lack of 'security-by-design' of legacy technologies. Consequently, it is also important to tackle the security issues at the field level as well as the IT level that have been highlighted by the complementary risk assessment study. This study reassured Cebi regarding the adoption of OPC-UA as an interconnection framework. Nowadays, OPC-UA is used at Cebi for publishing data to various applications.

The second part of the dissertation focuses on the improvement of the flexibility of the process controls. The data collection architecture allows for the use of data-driven approaches to deal with this volume of data. These data-driven approaches such as AI-based applications have several use cases at the factory level, but if those approaches could be implemented at the production cell level, they might help improve the flexibility of the process controls and consequently allow for more flexible production lines that could adapt to a changing environment and enable productivity gains. Our study investigates to what extent the integration of ML in the control architecture could help. Our intuition is that it is necessary to step away from the static PLC programs and replace them by ML methods, which could help the production lines to adapt themselves to an evolving environment and even allow the lines to self-configure. In order to conduct experiments in a safe testing environment without risking any downtime of the production of our partner Cebi and because at the time of the study the data collection infrastructure was not ready at Cebi, we chose to use a scale model of a factory. To ensure the realistic and representative nature of the experiments, we chose actual industrial PLCs to control the factory as well as new advanced PLCs so as to study their capabilities to integrate ML at the edge for various applications. We focus on a representative

Time Series Classification (TSC) problem one could encounter in the industry and highlight the implications and limitations of using such a method from a technical point of view. Our study show that the integration of traditional ML in the process control architecture allows for improvements in terms of production flexibility. Additionally, it shows that using new generations of PLCs with open communication capabilities allow for the use of ML at the edge, by using a ML server as an API for example, as long as the prediction delays are suitable for the application. This study paved the way for Cebi to start using ML in the control architecture, not for flexibility purposes yet, but for quality control.

The third part of the dissertation investigates a trustworthy data storage architecture for traceability purposes, enabled by the data collection architecture. *Product traceability* is another key feature of the Industry 4.0 and one of the biggest challenge in the manufacturing industry. Indeed, keeping track and trace of the different processes and operations during the product life cycle requires having a secure data collection infrastructure and is crucial to prevent high-cost recalls and provide the customers with precise information and circumstances when recalls happen. For this matter, blockchain and DLT are emerging in both the industrial world and the academic world. As a matter of fact, these technologies allow to store data in an immutable way, which is an important asset, especially when the system is shared between various participants and some of them are not trusted. This part is composed of three studies. In the first study, we identify and compare five of the major permissioned blockchain frameworks used in the industry, based on the literature. This study highlights several point for both researchers and practitioners. Indeed, the choice of a platform has to be a trade-off as there is no best platform regarding all the criteria. Additionally, there is a lack of comparison of the platforms in the same environment. However, the implementation of the blockchain platforms is not straightforward because of the number of parameters and the lack of guidance on this subject in the literature. This leads to the second study in which we propose a methodology for selecting the optimal blockchain configuration. We demonstrate the applicability of this methodology by instantiating it using Taguchi DoE method for defining the configurations and AHP combined with TOPSIS to weight and rank the configurations after experimenting with private Ethereum. Conducting this study allowed us to witness how time consuming and difficult it is to change the blockchain configurations and restart the experiments. To overcome this, in the third study we propose BlockPerf, a blockchain simulation/emulation framework covering as many blockchain layers as possible and that allows to conduct experiments and to implement configurations in a easier way since it does not require to implement the whole platform. But prior to this, the study proposes a literature review of the blockchain simulator, and a comparison of BlockPerf with BlockSim, which is a state of the art simula-

tor, and with a bitcoin implementation. Our study show that BlockPerf provides more realistic results than BlockSim (e.g., at the Node/Data and Network layers) respectively improved by 39% and 55% in average.

Overall, we could notice that once data is available, there are plenty of technical solutions to engage the Industry 4.0 paradigm shift and reach the required properties. However, in the context of the partnership with Cebi Luxembourg S.A., we could also witness the importance of the cultural factor in such a paradigm change. As a matter of fact, this transition brings together technologies that have been research topics for years as well as new technologies that have not yet been proven efficient and effective in the industry. This can create some reluctance within the enterprise. Testing these solutions can be difficult for the enterprises due to the lack of expertise or testing environment for such technologies. Delegating such experiments and tests to a third party with expertise on such technologies is costly and it might not even be sufficient for the enterprise to evaluate the return of investment. Additionally, adopting some of the technologies could induce shifts in some of the professions, for which the employees in the occupation of the professions in question need to receive training on the new technologies.

To sum up, the works included in this dissertation contribute to further understand the implications of using some of the technologies and the way they can be applied to progress towards Industry 4.0. But there are still challenges to be addressed, whether they are from a scientific perspective or a cultural perspective.

7.2 Perspectives

There are still research challenges to be addressed when looking at the studies conducted in this dissertation. Regarding the security of the data collection infrastructure, there are still directions to explore, for example, investigating the security mechanisms at the Layer 2 of the OSI model specified by the TSN working group. For instance, the project P802.1Qci (Per-Stream Filtering and Policing) that will enable to check whether a frame fits a reserved timeslot. This would enable to mitigate the impact of an abnormal traffic (message flooding/DoS attack or babbling idiot failure) or it could even enable to completely stop it (and quickly to come back to a normal situation). In addition, 802.1AEcg (MACsec) could be used for maintaining data confidentiality between different authorised systems (interconnected to an Ethernet network). By using it, all the frames are authenticated and (optionally) encrypted. It enables to prevent attacks such as eavesdropping or message replay. Of course, the overhead introduced in this mechanism need to be taken into account when defining the TSN timeslots.

Regarding the flexibility of the process control, we noticed that for some cases, the ML model needed to be retrained in order to classify accurately. The retraining

process was done manually. It could be interesting to automate the retraining process.

Regarding the the methodology for selecting the optimal blockchain configuration, it could be interesting to compare the instantiated methodology with another set of techniques, so as to compare the obtained ranking with the one we have.

Finally, regarding BlockPerf, the blockchain simulation/emulation framework, there are several improvements that can be made: i) the simulator at the moment only support the bitcoin blockchain platform, there is still some work to do for it to support other frameworks, ii) the contract layer is not yet supported so there is still some progress to make towards this and iii) the cost of deploying nodes over different geographical locations is still significant, so the simulator may not the perfect solution for reducing the cost of deploying blockchain platforms, there is still room for improvements regarding this.

8

Appendices

8.1 AHP computations

The pairwise comparison matrix of L1 level P_{L1} is in 8.1.

$$\mathbf{P_{L1}} = \begin{array}{c} \\ Node/Data \\ Appli. \end{array} \begin{array}{c} \begin{array}{cc} Node/Data & Appli. \end{array} \\ \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \end{array} \tag{8.1}$$

Then the weight vector W_{L1} can then be computed using equation 5.2. The calculation is detailed below in 8.2.

$$\begin{aligned} W_{Node/Data} &= \frac{1+1}{1+1+1+1} = \frac{1}{2} \\ W_{Appli.} &= \frac{1+1}{1+1+1+1} = \frac{1}{2} \\ W_{L1} &= [W_{Node/Data} \quad W_{Appli.}] = [\frac{1}{2} \quad \frac{1}{2}] \end{aligned} \tag{8.2}$$

In the same way, the two pairwise comparison matrices $P_{L2_{Node/Data}}$ and $P_{L2_{Appli.}}$ for L2 can be created as in 8.3 and 8.4.

$$\mathbf{P_{L2_{Node/Data}}} = \begin{array}{c} \\ (k) \\ (l_1) \\ (l_2) \end{array} \begin{array}{c} \begin{array}{ccc} (k) & (l_1) & (l_2) \end{array} \\ \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \end{array} \tag{8.3}$$

$$\mathbf{P_{L2_{Appli.}}} = \begin{array}{c} \\ (b_1) \\ (b_2) \end{array} \begin{array}{c} \begin{array}{cc} (b_1) & (b_2) \end{array} \\ \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \end{array} \tag{8.4}$$

Then the weight vectors for Node/Data and Application $W_{L2_{Node/Data}}$ and $W_{L2_{Appli.}}$ can be computed using equation 5.2 as in 8.5 and 8.6.

$$\begin{aligned} W_k &= \frac{1+1+1}{1+1+1+1+1+1} = \frac{1}{3} \\ W_{l_1} &= \frac{1+1+1}{1+1+1+1+1+1} = \frac{1}{3} \\ W_{l_2} &= \frac{1+1+1}{1+1+1+1+1+1} = \frac{1}{3} \\ W_{L2_{Node/Data}} &= [W_k \quad W_{l_1} \quad W_{l_2}] = [\frac{1}{3} \quad \frac{1}{3} \quad \frac{1}{3}] \end{aligned} \tag{8.5}$$

$$
\begin{aligned}
W_{b_1} &= \frac{1+1}{1+1+1+1} = \frac{1}{2} \\
W_{b_2} &= \frac{1+1}{1+1+1+1} = \frac{1}{2} \\
W_{L2_{Appli.}} &= [W_{b_1} \quad W_{b_2}] = [\frac{1}{2} \quad \frac{1}{2}]
\end{aligned}
\tag{8.6}
$$

Finally the weights of the alternatives regarding each criterion can be computed. There are 5 score-based comparison matrices (since there are 5 criteria) of size 12×12 (since there are 12 alternatives). The matrices are presented in 8.8, 8.9, 8.10, 8.11, 8.12. These matrices are based on the experiments results.

The values inside of the matrices are computed following 5.3 or 5.4 depending on the characteristic of the criteria, cost or benefit. In our case, (k) and (l_1) are benefit criteria (meaning we want to maximise them) while (l_2), (b_1) and (b_2) are cost criteria. As an example, 8.7 is the computation of $S_{L3_k}(2,1)$ in matrix 8.8 which is the weight of configuration 5 over configuration 1 with regards to the (k) criterion (number of transactions), configuration 1 has 1438 transactions and configuration 5 has 718:

$$
S_{L3_k}(2,1) = \frac{718}{1438} = 0.4993 \tag{8.7}
$$

$$\mathbf{S_{L3_k}} = \begin{array}{c} \\ Cfg.1 \\ Cfg.5 \\ Cfg.6 \\ Cfg.7 \\ Cfg.8 \\ Cfg.9 \\ Cfg.10 \\ Cfg.12 \\ Cfg.13 \\ Cfg.14 \\ Cfg.15 \\ Cfg.16 \end{array} \begin{array}{c} \begin{array}{cccccccccccc} Cfg.1 & Cfg.5 & Cfg.6 & Cfg.7 & Cfg.8 & Cfg.9 & Cfg.10 & Cfg.12 & Cfg.13 & Cfg.14 & Cfg.15 & Cfg.16 \end{array} \\ \begin{bmatrix} 1 & 2.003 & 0.1 & 0.3034 & 0.1 & 6.042 & 0.9101 & 0.03338 & 9.101 & 0.6042 & 0.06676 & 0.04534 \\ 0.4993 & 1 & 0.04993 & 0.1515 & 0.04993 & 3.017 & 0.4544 & 0.01667 & 4.544 & 0.3017 & 0.03333 & 0.02264 \\ 10 & 20.03 & 1 & 3.034 & 1 & 60.42 & 9.101 & 0.3338 & 91.01 & 6.042 & 0.6676 & 0.4534 \\ 3.296 & 6.602 & 0.3296 & 1 & 0.3296 & 19.92 & 3 & 0.11 & 30 & 1.992 & 0.2201 & 0.1494 \\ 10 & 20.03 & 1 & 3.034 & 1 & 60.42 & 9.101 & 0.3338 & 91.01 & 6.042 & 0.6676 & 0.4534 \\ 0.1655 & 0.3315 & 0.01655 & 0.05021 & 0.01655 & 1 & 0.1506 & 0.005525 & 1.506 & 0.1 & 0.01105 & 0.007504 \\ 1.099 & 2.201 & 0.1099 & 0.3333 & 0.1099 & 6.639 & 1 & 0.03668 & 10 & 0.6639 & 0.07335 & 0.04981 \\ 29.96 & 60 & 2.996 & 9.089 & 2.996 & 181 & 27.27 & 1 & 272.7 & 18.1 & 2 & 1.358 \\ 0.1099 & 0.2201 & 0.01099 & 0.03333 & 0.01099 & 0.6639 & 0.1 & 0.003668 & 1 & 0.06639 & 0.007335 & 0.004981 \\ 1.655 & 3.315 & 0.1655 & 0.5021 & 0.1655 & 10 & 1.506 & 0.05525 & 15.06 & 1 & 0.1105 & 0.07504 \\ 14.98 & 30 & 1.498 & 4.544 & 1.498 & 90.5 & 13.63 & 0.5 & 136.3 & 9.05 & 1 & 0.6791 \\ 22.06 & 44.18 & 2.206 & 6.692 & 2.206 & 133.3 & 20.07 & 0.7363 & 200.7 & 13.33 & 1.473 & 1 \end{bmatrix} \end{array} \tag{8.8}$$

$$\mathbf{S_{L3_{l_1}}} = \begin{array}{c} \\ Cfg.1 \\ Cfg.5 \\ Cfg.6 \\ Cfg.7 \\ Cfg.8 \\ Cfg.9 \\ Cfg.10 \\ Cfg.12 \\ Cfg.13 \\ Cfg.14 \\ Cfg.15 \\ Cfg.16 \end{array} \begin{array}{c} \begin{array}{cccccccccccc} Cfg.1 & Cfg.5 & Cfg.6 & Cfg.7 & Cfg.8 & Cfg.9 & Cfg.10 & Cfg.12 & Cfg.13 & Cfg.14 & Cfg.15 & Cfg.16 \end{array} \\ \begin{bmatrix} 1 & 7.989 & 7.945 & 7.989 & 8.079 & 23.97 & 23.97 & 23.57 & 110.6 & 110.6 & 110.6 & 102.7 \\ 0.1252 & 1 & 0.9945 & 1 & 1.011 & 3 & 3 & 2.951 & 13.85 & 13.85 & 13.85 & 12.86 \\ 0.1259 & 1.006 & 1 & 1.006 & 1.017 & 3.017 & 3.017 & 2.967 & 13.92 & 13.92 & 13.92 & 12.93 \\ 0.1252 & 1 & 0.9945 & 1 & 1.011 & 3 & 3 & 2.951 & 13.85 & 13.85 & 13.85 & 12.86 \\ 0.1238 & 0.9889 & 0.9834 & 0.9889 & 1 & 2.967 & 2.967 & 2.918 & 13.69 & 13.69 & 13.69 & 12.71 \\ 0.04172 & 0.3333 & 0.3315 & 0.3333 & 0.3371 & 1 & 1 & 0.9836 & 4.615 & 4.615 & 4.615 & 4.286 \\ 0.04172 & 0.3333 & 0.3315 & 0.3333 & 0.3371 & 1 & 1 & 0.9836 & 4.615 & 4.615 & 4.615 & 4.286 \\ 0.04242 & 0.3389 & 0.337 & 0.3389 & 0.3427 & 1.017 & 1.017 & 1 & 4.692 & 4.692 & 4.692 & 4.357 \\ 0.00904 & 0.07222 & 0.07182 & 0.07222 & 0.07303 & 0.2167 & 0.2167 & 0.2131 & 1 & 1 & 1 & 0.9286 \\ 0.00904 & 0.07222 & 0.07182 & 0.07222 & 0.07303 & 0.2167 & 0.2167 & 0.2131 & 1 & 1 & 1 & 0.9286 \\ 0.00904 & 0.07222 & 0.07182 & 0.07222 & 0.07303 & 0.2167 & 0.2167 & 0.2131 & 1 & 1 & 1 & 0.9286 \\ 0.009736 & 0.07778 & 0.07735 & 0.07778 & 0.07865 & 0.2333 & 0.2333 & 0.2295 & 1.077 & 1.077 & 1.077 & 1 \end{bmatrix} \end{array} \tag{8.9}$$

$$\mathbf{S_{L3_{l_2}}} = \begin{array}{c} \\ Cfg.1 \\ Cfg.5 \\ Cfg.6 \\ Cfg.7 \\ Cfg.8 \\ Cfg.9 \\ Cfg.10 \\ Cfg.12 \\ Cfg.13 \\ Cfg.14 \\ Cfg.15 \\ Cfg.16 \end{array} \begin{array}{c} \begin{array}{cccccccccccc} Cfg.1 & Cfg.5 & Cfg.6 & Cfg.7 & Cfg.8 & Cfg.9 & Cfg.10 & Cfg.12 & Cfg.13 & Cfg.14 & Cfg.15 & Cfg.16 \end{array} \\ \begin{bmatrix} 1 & 8.336 & 8.142 & 8.111 & 7.989 & 24.2 & 23.97 & 24.15 & 119.9 & 120.3 & 119.8 & 120.1 \\ 0.12 & 1 & 0.9768 & 0.973 & 0.9585 & 2.903 & 2.875 & 2.897 & 14.38 & 14.43 & 14.38 & 14.41 \\ 0.1228 & 1.024 & 1 & 0.9962 & 0.9813 & 2.972 & 2.944 & 2.966 & 14.72 & 14.77 & 14.72 & 14.75 \\ 0.1233 & 1.028 & 1.004 & 1 & 0.985 & 2.984 & 2.955 & 2.977 & 14.78 & 14.83 & 14.78 & 14.81 \\ 0.1252 & 1.043 & 1.019 & 1.015 & 1 & 3.029 & 3 & 3.023 & 15 & 15.05 & 15 & 15.04 \\ 0.04132 & 0.3445 & 0.3365 & 0.3352 & 0.3302 & 1 & 0.9905 & 0.998 & 4.953 & 4.969 & 4.952 & 4.964 \\ 0.04172 & 0.3478 & 0.3397 & 0.3384 & 0.3333 & 1.01 & 1 & 1.008 & 5.001 & 5.017 & 5 & 5.012 \\ 0.04141 & 0.3452 & 0.3372 & 0.3359 & 0.3308 & 1.002 & 0.9925 & 1 & 4.963 & 4.98 & 4.962 & 4.974 \\ 0.008343 & 0.06955 & 0.06793 & 0.06767 & 0.06666 & 0.2019 & 0.2 & 0.2015 & 1 & 1.003 & 0.9999 & 1.002 \\ 0.008316 & 0.06932 & 0.06771 & 0.06745 & 0.06644 & 0.2012 & 0.1993 & 0.2008 & 0.9967 & 1 & 0.9966 & 0.999 \\ 0.008344 & 0.06956 & 0.06794 & 0.06768 & 0.06667 & 0.2019 & 0.2 & 0.2015 & 1 & 1.003 & 1 & 1.002 \\ 0.008324 & 0.06939 & 0.06778 & 0.06752 & 0.06651 & 0.2014 & 0.1995 & 0.201 & 0.9977 & 1.001 & 0.9976 & 1 \end{bmatrix} \end{array} \tag{8.10}$$

$$\mathbf{S_{L3_{b_1}}} = \begin{array}{c} \\ Cfg.1 \\ Cfg.5 \\ Cfg.6 \\ Cfg.7 \\ Cfg.8 \\ Cfg.9 \\ Cfg.10 \\ Cfg.12 \\ Cfg.13 \\ Cfg.14 \\ Cfg.15 \\ Cfg.16 \end{array} \begin{array}{c} \begin{array}{cccccccccccc} Cfg.1 & Cfg.5 & Cfg.6 & Cfg.7 & Cfg.8 & Cfg.9 & Cfg.10 & Cfg.12 & Cfg.13 & Cfg.14 & Cfg.15 & Cfg.16 \end{array} \\ \begin{bmatrix} 1 & 1.333 & 9.467 & 3.307 & 15.45 & 0.9733 & 3.587 & 263.4 & 1.787 & 8.613 & 242 & 425.1 \\ 0.75 & 1 & 7.1 & 2.48 & 11.59 & 0.73 & 2.69 & 197.6 & 1.34 & 6.46 & 181.5 & 318.8 \\ 0.1056 & 0.1408 & 1 & 0.3493 & 1.632 & 0.1028 & 0.3789 & 27.83 & 0.1887 & 0.9099 & 25.56 & 44.9 \\ 0.3024 & 0.4032 & 2.863 & 1 & 4.673 & 0.2944 & 1.085 & 79.67 & 0.5403 & 2.605 & 73.17 & 128.5 \\ 0.06471 & 0.08628 & 0.6126 & 0.214 & 1 & 0.06299 & 0.2321 & 17.05 & 0.1156 & 0.5574 & 15.66 & 27.51 \\ 1.027 & 1.37 & 9.726 & 3.397 & 15.88 & 1 & 3.685 & 270.6 & 1.836 & 8.849 & 248.6 & 436.7 \\ 0.2788 & 0.3717 & 2.639 & 0.9219 & 4.309 & 0.2714 & 1 & 73.45 & 0.4981 & 2.401 & 67.46 & 118.5 \\ 0.003796 & 0.005061 & 0.03594 & 0.01255 & 0.05866 & 0.003695 & 0.01362 & 1 & 0.006782 & 0.0327 & 0.9185 & 1.614 \\ 0.5597 & 0.7463 & 5.299 & 1.851 & 8.649 & 0.5448 & 2.007 & 147.4 & 1 & 4.821 & 135.4 & 237.9 \\ 0.1161 & 0.1548 & 1.099 & 0.3839 & 1.794 & 0.113 & 0.4164 & 30.58 & 0.2074 & 1 & 28.09 & 49.35 \\ 0.004133 & 0.005511 & 0.03912 & 0.01367 & 0.06387 & 0.004023 & 0.01482 & 1.089 & 0.007384 & 0.0356 & 1 & 1.757 \\ 0.002353 & 0.003137 & 0.02227 & 0.007779 & 0.03636 & 0.00229 & 0.008438 & 0.6197 & 0.004203 & 0.02026 & 0.5692 & 1 \end{bmatrix} \end{array} \tag{8.11}$$

$$\mathbf{S_{L3_{b_2}}} = \begin{array}{c} \\ Cfg.1 \\ Cfg.5 \\ Cfg.6 \\ Cfg.7 \\ Cfg.8 \\ Cfg.9 \\ Cfg.10 \\ Cfg.12 \\ Cfg.13 \\ Cfg.14 \\ Cfg.15 \\ Cfg.16 \end{array} \begin{array}{c} \begin{array}{cccccccccccc} Cfg.1 & Cfg.5 & Cfg.6 & Cfg.7 & Cfg.8 & Cfg.9 & Cfg.10 & Cfg.12 & Cfg.13 & Cfg.14 & Cfg.15 & Cfg.16 \end{array} \\ \begin{bmatrix} 1 & 0.9167 & 2.5 & 2.25 & 4 & 0.4167 & 1.333 & 8.917 & 0.4167 & 1.667 & 12.83 & 21 \\ 1.091 & 1 & 2.727 & 2.455 & 4.364 & 0.4545 & 1.455 & 9.727 & 0.4545 & 1.818 & 14 & 22.91 \\ 0.4 & 0.3667 & 1 & 0.9 & 1.6 & 0.1667 & 0.5333 & 3.567 & 0.1667 & 0.6667 & 5.133 & 8.4 \\ 0.4444 & 0.4074 & 1.111 & 1 & 1.778 & 0.1852 & 0.5926 & 3.963 & 0.1852 & 0.7407 & 5.704 & 9.333 \\ 0.25 & 0.2292 & 0.625 & 0.5625 & 1 & 0.1042 & 0.3333 & 2.229 & 0.1042 & 0.4167 & 3.208 & 5.25 \\ 2.4 & 2.2 & 6 & 5.4 & 9.6 & 1 & 3.2 & 21.4 & 1 & 4 & 30.8 & 50.4 \\ 0.75 & 0.6875 & 1.875 & 1.688 & 3 & 0.3125 & 1 & 6.688 & 0.3125 & 1.25 & 9.625 & 15.75 \\ 0.1121 & 0.1028 & 0.2804 & 0.2523 & 0.4486 & 0.04673 & 0.1495 & 1 & 0.04673 & 0.1869 & 1.439 & 2.355 \\ 2.4 & 2.2 & 6 & 5.4 & 9.6 & 1 & 3.2 & 21.4 & 1 & 4 & 30.8 & 50.4 \\ 0.6 & 0.55 & 1.5 & 1.35 & 2.4 & 0.25 & 0.8 & 5.35 & 0.25 & 1 & 7.7 & 12.6 \\ 0.07792 & 0.07143 & 0.1948 & 0.1753 & 0.3117 & 0.03247 & 0.1039 & 0.6948 & 0.03247 & 0.1299 & 1 & 1.636 \\ 0.04762 & 0.04365 & 0.119 & 0.1071 & 0.1905 & 0.01984 & 0.06349 & 0.4246 & 0.01984 & 0.07937 & 0.6111 & 1 \end{bmatrix} \end{array} \tag{8.12}$$

Then the weight vectors of the alternatives regarding each criterion can be calculated using 5.5. Equation 8.13 show the weights of the alternatives with regards to the (k) criterion and W_k^A in 8.14 is the resulting weight vector. Equation 8.15 show the weights of the alternatives with regards to the (l_1) criterion and $W_{l_1}^A$ in 8.16 is the resulting weight vector. Equation 8.17 show the weights of the alternatives with regards to the (l_2) criterion and $W_{l_2}^A$ in 8.18 is the resulting weight vector. Equation 8.19 show the weights of the alternatives with regards to the (b_1) criterion and $W_{b_1}^A$ in 8.20 is the resulting weight vector. Equation 8.21 show the weights of the alternatives with regards to the (b_2) criterion and $W_{b_2}^A$ in 8.22 is the resulting weight vector.

$$
\begin{aligned}
W_k^{Cfg.1} &= 0.0105 \\
W_k^{Cfg.5} &= 0.0053 \\
W_k^{Cfg.6} &= 0.1055 \\
W_k^{Cfg.7} &= 0.0348 \\
W_k^{Cfg.8} &= 0.1055 \\
W_k^{Cfg.9} &= 0.0017 \\
W_k^{Cfg.10} &= 0.0116 \\
W_k^{Cfg.12} &= 0.3160 \\
W_k^{Cfg.13} &= 0.0012 \\
W_k^{Cfg.14} &= 0.0175 \\
W_k^{Cfg.15} &= 0.1580 \\
W_k^{Cfg.16} &= 0.2326
\end{aligned}
\tag{8.13}
$$

$$
W_k^A = \begin{bmatrix} 0.0105 \\ 0.0053 \\ 0.1055 \\ 0.0348 \\ 0.1055 \\ 0.0017 \\ 0.0116 \\ 0.3160 \\ 0.0012 \\ 0.0175 \\ 0.1580 \\ 0.2326 \end{bmatrix} \tag{8.14}
$$

$$
\begin{aligned}
W_{l_1}^{Cfg.1} &= 0.6014 \\
W_{l_1}^{Cfg.5} &= 0.0753 \\
W_{l_1}^{Cfg.6} &= 0.0757 \\
W_{l_1}^{Cfg.7} &= 0.0753 \\
W_{l_1}^{Cfg.8} &= 0.0744 \\
W_{l_1}^{Cfg.9} &= 0.0251 \\
W_{l_1}^{Cfg.10} &= 0.0251 \\
W_{l_1}^{Cfg.12} &= 0.0255 \\
W_{l_1}^{Cfg.13} &= 0.0054 \\
W_{l_1}^{Cfg.14} &= 0.0054 \\
W_{l_1}^{Cfg.15} &= 0.0054 \\
W_{l_1}^{Cfg.16} &= 0.0059
\end{aligned}
\tag{8.15}
$$

$$
W_{l_1}^{A} = \begin{bmatrix} 0.6014 \\ 0.0753 \\ 0.0757 \\ 0.0753 \\ 0.0744 \\ 0.0251 \\ 0.0251 \\ 0.0255 \\ 0.0054 \\ 0.0054 \\ 0.0054 \\ 0.0059 \end{bmatrix} \tag{8.16}
$$

$$
\begin{aligned}
W_{l_2}^{Cfg.1} &= 0.6064 \\
W_{l_2}^{Cfg.5} &= 0.0728 \\
W_{l_2}^{Cfg.6} &= 0.0745 \\
W_{l_2}^{Cfg.7} &= 0.0748 \\
W_{l_2}^{Cfg.8} &= 0.0759 \\
W_{l_2}^{Cfg.9} &= 0.0251 \\
W_{l_2}^{Cfg.10} &= 0.0253 \\
W_{l_2}^{Cfg.12} &= 0.0251 \\
W_{l_2}^{Cfg.13} &= 0.0051 \\
W_{l_2}^{Cfg.14} &= 0.0050 \\
W_{l_2}^{Cfg.15} &= 0.0051 \\
W_{l_2}^{Cfg.16} &= 0.0050
\end{aligned}
\tag{8.17}
$$

$$
W_{l_2}^{A} = \begin{bmatrix} 0.6064 \\ 0.0728 \\ 0.0745 \\ 0.0748 \\ 0.0759 \\ 0.0251 \\ 0.0253 \\ 0.0251 \\ 0.0051 \\ 0.0050 \\ 0.0051 \\ 0.0050 \end{bmatrix} \tag{8.18}
$$

$$
\begin{aligned}
W_{b_1}^{Cfg.1} &= 0.2372 \\
W_{b_1}^{Cfg.5} &= 0.1779 \\
W_{b_1}^{Cfg.6} &= 0.0251 \\
W_{b_1}^{Cfg.7} &= 0.0717 \\
W_{b_1}^{Cfg.8} &= 0.0154 \\
W_{b_1}^{Cfg.9} &= 0.2437 \\
W_{b_1}^{Cfg.10} &= 0.0661 \\
W_{b_1}^{Cfg.12} &= 0.0009 \\
W_{b_1}^{Cfg.13} &= 0.1328 \\
W_{b_1}^{Cfg.14} &= 0.0275 \\
W_{b_1}^{Cfg.15} &= 0.0009 \\
W_{b_1}^{Cfg.16} &= 0.0005
\end{aligned}
\tag{8.19}
$$

$$
W_{b_1}^{A} = \begin{bmatrix} 0.2372 \\ 0.1779 \\ 0.0251 \\ 0.0717 \\ 0.0154 \\ 0.2437 \\ 0.0661 \\ 0.0009 \\ 0.1328 \\ 0.0275 \\ 0.0009 \\ 0.0005 \end{bmatrix} \tag{8.20}
$$

$$
\begin{aligned}
W_{b_2}^{Cfg.1} &= 0.1045 \\
W_{b_2}^{Cfg.5} &= 0.1140 \\
W_{b_2}^{Cfg.6} &= 0.0418 \\
W_{b_2}^{Cfg.7} &= 0.0464 \\
W_{b_2}^{Cfg.8} &= 0.0261 \\
W_{b_2}^{Cfg.9} &= 0.2507 \\
W_{b_2}^{Cfg.10} &= 0.0783 \\
W_{b_2}^{Cfg.12} &= 0.0117 \\
W_{b_2}^{Cfg.13} &= 0.2507 \\
W_{b_2}^{Cfg.14} &= 0.0627 \\
W_{b_2}^{Cfg.15} &= 0.0081 \\
W_{b_2}^{Cfg.16} &= 0.0050
\end{aligned}
\tag{8.21}
$$

$$
W_{b_2}^{A} = \begin{bmatrix} 0.1045 \\ 0.1140 \\ 0.0418 \\ 0.0464 \\ 0.0261 \\ 0.2507 \\ 0.0783 \\ 0.0117 \\ 0.2507 \\ 0.0627 \\ 0.0081 \\ 0.0050 \end{bmatrix} \tag{8.22}
$$

8.2 TOPSIS computations

8.23 presents the resulting matrix. In order to clarify the computation, let us detail the calculation of $GW_C^A(1,1)$ and $GW_C^A(1,2)$ respectively in equations 8.24

and 8.25.

$$\mathbf{GW_C^A} = \begin{bmatrix} 0.001758 & 0.1002 & 0.1011 & 0.05931 & 0.02611 \\ 0.0008776 & 0.01255 & 0.01213 & 0.04448 & 0.02849 \\ 0.01758 & 0.01262 & 0.01241 & 0.006265 & 0.01045 \\ 0.005794 & 0.01255 & 0.01246 & 0.01794 & 0.01161 \\ 0.01758 & 0.01241 & 0.01265 & 0.003838 & 0.006529 \\ 0.0002909 & 0.004182 & 0.004177 & 0.06094 & 0.06268 \\ 0.001931 & 0.004182 & 0.004217 & 0.01654 & 0.01959 \\ 0.05266 & 0.004252 & 0.004185 & 0.0002252 & 0.002929 \\ 0.0001931 & 0.0009062 & 0.0008433 & 0.0332 & 0.06268 \\ 0.002909 & 0.0009062 & 0.0008405 & 0.006886 & 0.01567 \\ 0.02633 & 0.0009062 & 0.0008434 & 0.0002451 & 0.002035 \\ 0.03877 & 0.0009759 & 0.0008413 & 0.0001395 & 0.001244 \end{bmatrix} \tag{8.23}$$

$$GW_C^A(1,1) = W_k^A(1) \times W_k \times W_{Node/Data} = 0.0105 \times \frac{1}{3} \times \frac{1}{2} = 0.001758 \tag{8.24}$$

$$GW_C^A(1,2) = W_{l_1}^A(1) \times W_k \times W_{Node/Data} = 0.6014 \times \frac{1}{3} \times \frac{1}{2} = 0.1002 \tag{8.25}$$

Using equations 5.7 and 5.8, the best possible solution d^+ and the worst d^- can be computed. Results are in 8.26 and 8.27.

$$d^+ = max(GW_C^A) = 0.1011 \tag{8.26}$$

$$d^- = min(GW_C^A) = 0.0001395 \tag{8.27}$$

Then the distances $D^+(A_l)$ and $D^-(A_l)$ of the alternatives to respectively the best solution d^+ and to the worst solution d^- can be computed using equations 5.9 and 5.10. 8.28 and 8.29 show the resulting vectors containing the distances of each alternative to best and worst solution.

$$D^{+}(A) = \begin{bmatrix} 0.2169 \\ 0.4068 \\ 0.4460 \\ 0.4450 \\ 0.4523 \\ 0.3731 \\ 0.4589 \\ 0.4411 \\ 0.4075 \\ 0.4781 \\ 0.4750 \\ 0.4634 \end{bmatrix} \tag{8.28}$$

$$D^{-}(A) = \begin{bmatrix} 0.2878 \\ 0.0978 \\ 0.0586 \\ 0.0596 \\ 0.0523 \\ 0.1316 \\ 0.0458 \\ 0.0636 \\ 0.0971 \\ 0.0256 \\ 0.0297 \\ 0.0413 \end{bmatrix} \tag{8.29}$$

After computing the weights for criteria at all levels (L1, L2, L3), the next step is to compute the global weight matrix as input of the TOPSIS method using equation 5.6. Finally, the score of the alternatives $R(A_l)$ with regards to all the combined criteria that leads to the ranking can be calculated using equation 5.11. 8.30 gives the score of each alternative.

$$R(A) = \begin{bmatrix} 0.5703 \\ 0.1938 \\ 0.1162 \\ 0.1182 \\ 0.1036 \\ 0.2607 \\ 0.0907 \\ 0.1259 \\ 0.1924 \\ 0.0525 \\ 0.0588 \\ 0.0818 \end{bmatrix} \tag{8.30}$$

List of publications

Papers included in the dissertation

- [PRLT19]: *J. Polge, J. Robert and Y. Le Traon, "Assessing the impact of attacks on OPC-UA applications in the Industry 4.0 era," 2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC), 2019, pp. 1-6, doi: 10.1109/CCNC.2019.8651671.*
- [PRLT20b]: *J. Polge, J. Robert, Y. Le Traon, "A Case Driven Study of the Use of Time Series Classification for Flexibility in Industry 4.0". Sensors 2020, 20, 7273. doi: 10.3390/s20247273*
- [PGK+21]: *J. Polge, J. Robert, Y. Le Traon, "Permissioned blockchain frameworks in the industry: A comparison", ICT Express, Volume 7, Issue 2, 2021, pp. 229-233, ISSN 2405-9595, doi: 10.1016/j.icte.2020.09.002.*
- [PRL21]: *J. Polge, S. Ghatpande, S. Kubler, J. Robert and Y. Le Traon, "BlockPerf: A Hybrid Blockchain Emulator/Simulator Framework," in IEEE Access, vol. 9, pp. 107858-107872, 2021, doi: 10.1109/ACCESS.2021.3101044.*

List of figures

List of tables

List of algorithms

Bibliography

[AAKS17] Abdul Wahab Ahmed, Mian Muhammad Ahmed, Omair Ahmad Khan, and Munam Ali Shah. A comprehensive analysis on the security threats and their countermeasures of iot. *INTERNATIONAL JOURNAL OF ADVANCED COMPUTER SCIENCE AND APPLICATIONS*, 8(7):489–501, 2017.

[AB02] R. Albert and A.-L. Barabási. Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47, 2002.

[ABB+18] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, et al. Hyperledger fabric: a distributed operating system for permissioned blockchains. In *Proceedings of the Thirteenth EuroSys Conference*, page 30. ACM, 2018.

[ABdSR14] Thiago Rodrigues Alves, Mario Buratto, Flavio Mauricio de Souza, and Thelma Virginia Rodrigues. Openplc: An open source alternative to automation. In *IEEE Global Humanitarian Technology Conference (GHTC 2014)*, pages 585–589, 2014.

[ABK+20] Nandini Agrawal, Osman Biçer, Alptekin Küpçü, et al. Blocksim-net: A network based blockchain simulator. *arXiv preprint arXiv:2011.03241*, 2020.

[AHL+16] H Arasteh, V Hosseinnezhad, V Loia, A Tommasetti, O Troisi, M Shafie-Khah, and P Siano. Iot-based smart cities: a survey. In *2016 IEEE 16th International Conference on Environment and Electrical Engineering (EEEIC)*, pages 1–6. IEEE, 2016.

[AJM19] J. Al-Jaroodi and N. Mohamed. Blockchain in industries: A survey. *IEEE Access*, 7:36500–36515, 2019.

[AJS19] Joe Abou Jaoude and Raafat George Saade. Blockchain applications–usage in different domains. *IEEE Access*, 7:45360–45381, 2019.

[AKR+13] Elli Androulaki, Ghassan O Karame, Marc Roeschlin, Tobias Scherer, and Srdjan Capkun. Evaluating user privacy in bitcoin. In *International Conference on Financial Cryptography and Data Security*, pages 34–51. Springer, 2013.

[AM14] Ahmad W Atamli and Andrew Martin. Threat-based security analysis for the internet of things. In *Secure Internet of Things (SIoT), 2014 International Workshop on*, pages 35–43. IEEE, 2014.

[AMY+18] Alireza Abdoli, Amy C Murillo, Chin-Chia M Yeh, Alec C Gerry, and Eamonn J Keogh. Time series classification to improve poultry welfare. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 635–642. IEEE, 2018.

[AOK+19] Yusuke Aoki, Kai Otsuki, Takeshi Kaneko, Ryohei Banno, and Kazuyuki Shudo. Simblock: A blockchain network simulator. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 325–329. IEEE, 2019.

[ARG+19] Nitish Andola, Raghav, Manas Gogoi, S. Venkatesan, and Shekhar Verma. Vulnerabilities on hyperledger fabric. *Pervasive and Mobile Computing*, 59:101050, 2019.

[ATS] José Alonso-Tovar and Baidya Nath Saha. Bayesian network classifier with efficient statistical time-series features for the classification of robot execution failures.

[AvM19] Maher Alharby and Aad van Moorsel. Blocksim: A simulation framework for blockchain systems. *SIGMETRICS Perform. Eval. Rev.*, 46(3):135–138, January 2019.

[BC94] Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, 1994.

[BCMC08] José Barata, Luís Camarinha-Matos, and Gonçalo Cândido. A multiagent-based control system applied to an educational

shop floor. *Robotics and Computer-Integrated Manufacturing*, 24(5):597–605, 2008.

[BCMM17] Mario Luca Bernardi, Marta Cimitile, Fabio Martinelli, and Francesco Mercaldo. A time series classification approach to game bot detection. In *Proceedings of the 7th International Conference on Web Intelligence, Mining and Semantics*, pages 1–11, 2017.

[BCMM18] Mario Luca Bernardi, Marta Cimitile, Fabio Martinelli, and Francesco Mercaldo. Driver and path detection through time-series classification. *Journal of Advanced Transportation*, 2018, 2018.

[BDFN+16] Miguel Angel Ortiz Barrios, Fabio De Felice, Kevin Parra Negrete, Brandon Aleman Romero, Adriana Yaruro Arenas, and Antonella Petrillo. An ahp-topsis integrated model for selecting the most appropriate tomography equipment. *International Journal of Information Technology & Decision Making*, 15(04):861–885, 2016.

[BLB+17] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31(3):606–660, 2017.

[BLHB15] Anthony Bagnall, Jason Lines, Jon Hills, and Aaron Bostrom. Time-series classification with cote: the collective of transformation-based ensembles. *IEEE Transactions on Knowledge and Data Engineering*, 27(9):2522–2535, 2015.

[BMR+20] Theodor Borangiu, Octavian Morariu, Silviu Raileanu, Damien Trentesaux, Paulo LEIT AO, and Jose Barata. Digital transformation of manufacturing. industry of the future with cyber-physical production systems. *SCIENCE AND TECHNOLOGY*, 23(1):3–37, 2020.

[BMZ18] L. M. Bach, B. Mihaljevic, and M. Zagar. Comparative analysis of blockchain consensus algorithms. In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1545–1550, 2018.

[BPP+19] Anthony Brunel, Johanna Pasquet, Jérôome PASQUET, Nancy Rodriguez, Frédéric Comby, Dominique Fouchez, and Marc Chaumont. A cnn adapted to time series for the classification of supernovae. *Electronic Imaging*, 2019(14):90–1, 2019.

[BRLT21] Paul-Lou Benedick, Jérémy Robert, and Yves Le Traon. A systematic approach for evaluating artificial intelligence models in industrial settings. *Sensors*, 21(18), 2021.

[BRT19] Paul-Lou Benedick, Jérémy Robert, and Yves Le Traon. Trident: A three-steps strategy to digitise an industrial system for stepping into industry 4.0. In *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*, volume 1, pages 3037–3042, 2019.

[BS19] Ryohei Banno and Kazuyuki Shudo. Simulating a blockchain network with simblock. In *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 3–4. IEEE, 2019.

[BSKC18] Arati Baliga, I Subhod, Pandurang Kamat, and Siddhartha Chatterjee. Performance evaluation of the quorum blockchain platform. *arXiv preprint arXiv:1809.03421*, 2018.

[BSV+18] Arati Baliga, Nitesh Solanki, Shubham Verekar, Amol Pednekar, Pandurang Kamat, and Siddhartha Chatterjee. Performance characterization of hyperledger fabric. In *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, pages 65–74. IEEE, 2018.

[BTJ20] Muhammad Babar, Muhammad Usman Tariq, and Mian Ahmad Jan. Secure and resilient demand side management engine using machine learning for iot-enabled smart grid. *Sustainable Cities and Society*, 62:102370, 2020.

[BTP+20] Umesh Bodkhe, Sudeep Tanwar, Karan Parekh, Pimal Khanpara, Sudhanshu Tyagi, Neeraj Kumar, and Mamoun Alazab. Blockchain for industry 4.0: A comprehensive review. *IEEE Access*, 8:79764–79800, 2020.

[CCM10] Salvatore Cavalieri, Giovanni Cutuli, and Salvatore Monteleone. Evaluating impact of security on opc ua performance. In *Human System Interactions (HSI), 2010 3rd Conference on*, pages 687–694. IEEE, 2010.

[Com16] International Electrotechnical Commission. Iec 62541-1 - opc unified architecture - part 1: Overview and concepts. Technical report, 2016.

[cor] The corda platform: an introduction. https://www.r3.com/wp-content/uploads/2019/06/corda-platform-whitepaper.pdf. Last accessed 31/03/2020.

[CRFAF17] Sujit Rokka Chhetri, Nafiul Rashid, Sina Faezi, and Mohammad Abdullah Al Faruque. Security trends and advances in manufacturing systems in the era of industry 4.0. In *IEEE/ACM international conference on computer aided design Google Scholar*, 2017.

[CSTK07] Mei-Tai Chu, Joseph Shyu, Gwo-Hshiung Tzeng, and Rajiv Khosla. Comparison among three analytical methods for knowledge communities group-decision analysis. *Expert systems with applications*, 33(4):1011–1024, 2007.

[CXZ+20] Longting Chen, Guanghua Xu, Sicong Zhang, Wenqiang Yan, and Qingqiang Wu. Health indicator construction of machinery based on end-to-end trainable convolution recurrent neural networks. *Journal of Manufacturing Systems*, 54:1–11, 2020.

[CYT20] Zi Hau Chin, Timothy Tzen Vun Yap, and Ian KT Tan. Simulating difficulty adjustment in blockchain with simblock. In *Proceedings of the 2nd ACM International Symposium on Blockchain and Secure Critical Infrastructure*, pages 192–197, 2020.

[DNJ18] Aditya Deshpande, Pezhman Nasirifard, and Hans-Arno Jacobsen. evibes: Configurable and interactive ethereum blockchain simulation framework. In *Proceedings of the 19th International Middleware Conference (Posters)*, pages 11–12, 2018.

[dsl] Dslf: Distributed ledger simulation framework. https://github.com/i13-msrg/dlsf. Accessed: 2021-01-25.

[DTS+08] Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, and Eamonn Keogh. Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 1(2):1542–1552, 2008.

[DWC+17] Tien Tuan Anh Dinh, Ji Wang, Gang Chen, Rui Liu, Beng Chin Ooi, and Kian-Lee Tan. Blockbench: A framework for analyzing private blockchains. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1085–1100, 2017.

[EA12] Philippe Esling and Carlos Agon. Time-series data mining. *ACM Computing Surveys (CSUR)*, 45(1):12, 2012.

[eth] A next-generation smart contract and decentralized application platform. `https://github.com/ethereum/wiki/wiki/White-Paper`. Last accessed 13/04/2021.

[evi] evibes plasma simulator. `https://github.com/i13-msrg/evibes-plasma`. Accessed: 2021-01-25.

[FAB+18] Simon N Foley, Fabien Autrel, Edwin Bourget, Thomas Cledel, Stephane Grunenwald, Jose Rubio Hernan, Alexandre Kabil, Raphael Larsen, Vivien M Rooney, and Kirsten Vanhulst. Science hackathons for cyberphysical system security research: Putting cps testbed platforms to good use. In *Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and PrivaCy*, pages 102–107. ACM, 2018.

[FBVI17] Syed Naeem Firdous, Zubair Baig, Craig Valli, and Ahmed Ibrahim. Modelling and evaluation of malicious attacks against the iot mqtt protocol. In *Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), 2017 IEEE International Conference on*, pages 748–755. IEEE, 2017.

[FC19] Carlos Faria and Miguel Correia. Blocksim: Blockchain simulator. In *2019 IEEE International Conference on Blockchain (Blockchain)*, pages 439–446. IEEE, 2019.

[FGKM20] C. Fan, S. Ghaemi, H. Khazaei, and P. Musilek. Performance evaluation of blockchain systems: A systematic survey. *IEEE Access*, 8:126927–126950, 2020.

[fISB17] Federal Office for Information Security (BSI). Opc ua security analysis. Technical report, March 2017.

[FJ14] Ben D Fulcher and Nick S Jones. Highly comparative feature-based time-series classification. *IEEE Transactions on Knowledge and Data Engineering*, 26(12):3026–3037, 2014.

[FK12] Andreas Fernbach and Wolfgang Kastner. Certificate management in opc ua applications: An evaluation of different trust models. In *Emerging Technologies & Factory Automation (ETFA), 2012 IEEE 17th Conference on*, pages 1–6. IEEE, 2012.

[FKM17] Johannes Fütterer, Maksymilian Kochanski, and Dirk Müller. Application of selected supervised learning methods for time series classification in building automation and control systems. *Energy Procedia*, 122:943–948, 2017.

[FMF20] Seyed Mehdi Fattahi, Adetokunbo Makanju, and Amin Milani Fard. Simba: An efficient simulator for blockchain applications. In *2020 50th Annual IEEE-IFIP International Conference on Dependable Systems and Networks-Supplemental Volume (DSN-S)*, pages 51–52. IEEE, 2020.

[for] Forbes: Blockchain 50 2019. `https://www.forbes.com/sites/michaeldelcastillo/2020/02/19/blockchain-50/`. Last accessed 31/03/2020.

[Fou02] OPC Foundation. Opc data access, 2002.

[GKLS20] F. Gräbe, N. Kannengiesser, S. Lins, and A. Sunyaev. Do not be fooled: Toward a holistic comparison of distributed ledger technology designs. In *Proceedings of the 53rd Hawaii International Conference on System Sciences*, 2020.

[GKW+16] Arthur Gervais, Ghassan O Karame, Karl Wüst, Vasileios Glykantzis, Hubert Ritzdorf, and Srdjan Capkun. On the security and performance of proof of work blockchains. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 3–16, 2016.

[Gos17] Sneha Goswami. Scalability analysis of blockchains through blockchain simulation. 2017.

[GWJ17] Yang Guo, Zhenyu Wu, and Yang Ji. A hybrid deep representation learning model for time series classification and prediction. In *2017 3rd International Conference on Big Data Computing and Communications (BIGCOM)*, pages 226–231. IEEE, 2017.

[hiv] Hive simulation tool. https://github.com/ethereum/hive. Accessed: 2021-01-25.

[HKZ+21] Huawei Huang, Wei Kong, Sicong Zhou, Zibin Zheng, and Song Guo. A survey of state-of-the-art on blockchains: Theories, modelings, and tools. *ACM Computing Surveys (CSUR)*, 54(2):1–42, 2021.

[HLD+18] Y. Hao, Y. Li, X. Dong, L. Fang, and P. Chen. Performance analysis of consensus algorithm in private blockchain. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 280–285, June 2018.

[HMM+] Hermann Haskamp, Michael Meyer, Romina Möllmann, Florian Orth, and Armando Walter Colombo. Benchmarking of existing opc ua implementations for industrie 4.0-compliant digitalization solutions.

[HSGX19] Runchao Han, Gary Shapiro, Vincent Gramoli, and Xiwei Xu. On the performance of distributed ledgers for internet of things. *Internet of Things*, page 100087, 2019.

[HY17] Jeonghan Hong and Junho Yoon. Multivariate time-series classification of sleep patterns using a hybrid deep learning architecture. In *2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom)*, pages 1–6. IEEE, 2017.

[ISO20] Blockchain and distributed ledger technologies - Reference architecture. Standard (under development), International Organization for Standardization, Geneva, CH, April 2020.

[itu] Technical paper hstp.dlt-rf: Distributed ledger technology: Regulatory framework. https://www.itu.int/dms_pub/itu-t/opb/tut/T-TUT-DLT-2019-RF-PDF-E.pdf. Accessed: 2021-04-12.

[Jaz14] Nasser Jazdi. Cyber physical systems in the context of industry 4.0. In *2014 IEEE international conference on automation, quality and testing, robotics*, pages 1–4. IEEE, 2014.

[JVSV22] Ajith Tom James, Dhruval Vaidya, Mustufa Sodawala, and Sunny Verma. Selection of bus chassis for large fleet operators

in india: An ahp-topsis approach. *Expert Syst. Appl.*, 186(C), dec 2022.

[JW18] M. A. Javarone and C. S. Wright. From bitcoin to bitcoin cash: a network analysis. In *Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems*, pages 77–81, 2018.

[KPGR19] Murat Kuzlu, Manisa Pipattanasomporn, Levent Gurses, and Saifur Rahman. Performance analysis of a hyperledger fabric blockchain framework: Throughput, latency and scalability. In *2019 IEEE International Conference on Blockchain (Blockchain)*, pages 536–540. IEEE, 2019.

[KRD+16] Sylvain Kubler, Jérémy Robert, William Derigent, Alexandre Voisin, and Yves Le Traon. A state-of the-art survey & testbed of fuzzy ahp (fahp) applications. *Expert Systems with Applications*, 65:398–422, 2016.

[KYH08] Yiyo Kuo, Taho Yang, and Guan-Wei Huang. The use of a grey-based taguchi method for optimizing multi-response simulation problems. *Engineering Optimization*, 40(6):517–528, 2008.

[LB15] Jason Lines and Anthony Bagnall. Time series classification with ensembles of elastic distance measures. *Data Mining and Knowledge Discovery*, 29(3):565–592, 2015.

[LDHB12] Jason Lines, Luke M Davis, Jon Hills, and Anthony Bagnall. A shapelet transform for time series classification. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 289–297. ACM, 2012.

[LFK+14] Heiner Lasi, Peter Fettke, Hans-Georg Kemper, Thomas Feld, and Michael Hoffmann. Industry 4.0. *Business & information systems engineering*, 6(4):239–242, 2014.

[Li18] Daoyuan Li. *Transforming Time Series for Efficient and Accurate Classification*. PhD thesis, University of Luxembourg, Luxembourg, Luxembourg, 2018.

[LKG21] G. Leduc, S. Kubler, and J.-P. Georges. Innovative blockchain-based farming marketplace and smart contract performance evaluation. *Journal of Cleaner Production*, 306:127055, 2021.

[LKL12] Jessica Lin, Rohan Khade, and Yuan Li. Rotation-invariant similarity in time series using bag-of-patterns representation. *Journal of Intelligent Information Systems*, 39(2):287–315, 2012.

[LKWL07] Jessica Lin, Eamonn Keogh, Li Wei, and Stefano Lonardi. Experiencing sax: a novel symbolic representation of time series. *Data Mining and knowledge discovery*, 15(2):107–144, 2007.

[LL15] In Lee and Kyoochun Lee. The internet of things (iot): Applications, investments, and challenges for enterprises. *Business Horizons*, 58(4):431–440, 2015.

[LLZS18] Changqing Liu, Yingguang Li, Guanyan Zhou, and Weiming Shen. A sensor fusion and support vector machine based approach for recognition of complex machining conditions. *Journal of Intelligent Manufacturing*, 29(8):1739–1752, 2018.

[LNJ18] Mohamed Riswan Abdul Lathif, Pezhman Nasirifard, and Hans-Arno Jacobsen. Cidds: A configurable and distributed dag-based distributed ledger simulation framework. In *Proceedings of the 19th International Middleware Conference (Posters)*, pages 7–8, 2018.

[LTB16] Jason Lines, Sarah Taylor, and Anthony Bagnall. Hive-cote: The hierarchical vote collective of transformation-based ensembles for time series classification. In *2016 IEEE 16th international conference on data mining (ICDM)*, pages 1041–1046. IEEE, 2016.

[LWT+18] Chengbao Liu, Xuelei Wang, Jie Tan, Lianjing Wang, Wei Sun, and Wenxiang He. Discharge voltage time series classification of lithium-ion cells based on deep neural networks. In *2018 IEEE 4th International Conference on Computer and Communications (ICCC)*, pages 2128–2132. IEEE, 2018.

[LZSW19] Zhixiong Li, Ziyang Zhang, Junchuan Shi, and Dazhong Wu. Prediction of surface roughness in extrusion-based additive manufacturing with machine learning. *Robotics and Computer-Integrated Manufacturing*, 57:488–495, 2019.

[MA17] Davor Maček and Dino Alagić. Comparisons of bitcoin cryptosystem with other common internet transaction systems by ahp technique. *Journal of Information and Organizational Sciences*, 41(1):69–87, 2017.

[MATR19] Giovanna Martínez-Arellano, German Terrazas, and Svetan Ratchev. Tool wear classification using time series imaging and deep learning. *The International Journal of Advanced Manufacturing Technology*, 104(9-12):3647–3662, 2019.

[MJ09] Alberto Montresor and Márk Jelasity. Peersim: A scalable p2p simulator. In *2009 IEEE Ninth International Conference on Peer-to-Peer Computing*, pages 99–100. IEEE, 2009.

[MJ15] Andrew Miller and Rob Jansen. Shadow-bitcoin: Scalable simulation via direct execution of multi-threaded applications. In *8th Workshop on Cyber Security Experimentation and Test ({CSET} 15)*, 2015.

[MKCA20] Parisa Mojaver, Shahram Khalilarya, Ata Chitsaz, and Mohsen Assadi. Multi-objective optimization of a power generation system based sofc using taguchi/ahp/topsis triple method. *Sustainable Energy Technologies and Assessments*, 38:100674, 2020.

[MLE+17] Nijat Mehdiyev, Johannes Lahann, Andreas Emrich, David Enke, Peter Fettke, and Peter Loos. Time series classification using deep learning for process planning: a case from the process industry. *Procedia Computer Science*, 114:242–249, 2017.

[MMZ16] Ahmed Mosallam, Kamal Medjaher, and Noureddine Zerhouni. Data-driven prognostic method based on bayesian approaches for direct remaining useful life prediction. *Journal of Intelligent Manufacturing*, 27(5):1037–1048, 2016.

[MOJH04] Saeed Maghsoodloo, Guttekin Ozdemir, Victoria Jordan, and Chen-Hsiu Huang. Strengths and limitations of taguchi's contributions to quality, manufacturing, and process engineering. *Journal of Manufacturing systems*, 23(2):73–126, 2004.

[MRR+17] Giuseppe Manco, Ettore Ritacco, Pasquale Rullo, Lorenzo Gallucci, Will Astill, Dianne Kimber, and Marco Antonelli. Fault detection and explanation through big data analysis on sensor streams. *Expert Systems with Applications*, 87:141–156, 2017.

[MSA19] A. A. Monrat, O. Schelén, and K. Andersson. A survey of blockchain from the perspectives of applications, challenges, and opportunities. *IEEE Access*, 7:117134–117151, 2019.

[MSF19] Zhenjie Ma, Ke Shi, and Kang Feng. A self-learning classification framework for industrial time series streams. *IFAC-PapersOnLine*, 52(13):1531–1536, 2019.

[MSH19] S. Maranhão, J.-. Seigneur, and R. Hu. Towards a standard to assess blockchain & other dlt platforms. In *ITU*, 2019.

[MSS18] Salome Maro, Jan-Philipp Steghöfer, and Miroslaw Staron. Software traceability in the automotive domain: Challenges and solutions. *Journal of Systems and Software*, 141:85–110, 2018.

[mul] Multichain private blockchain - white paper. `https://www.multichain.com/download/MultiChain-White-Paper.pdf`. Last accessed 31/03/2020.

[MVB19] Matthew Middlehurst, William Vickers, and Anthony Bagnall. Scalable dictionary classifiers for time series classification. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 11–19. Springer, 2019.

[MXZ+17] D. Mingxiao, M. Xiaofeng, Z. Zhe, W. Xiangwei, and C. Qijun. A review on consensus algorithm of blockchain. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 2567–2572, 2017.

[Nak08] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[NAR+20] Rizwan Ali Naqvi, Muhammad Arsalan, Abdul Rehman, Ateeq Ur Rehman, Woong-Kee Loh, and Anand Paul. Deep learning-based drivers emotion classification system in time series data for remote applications. *Remote Sensing*, 12(3):587, 2020.

[NBH+11] M. Naphade, G. Banavar, C. Harrison, J. Paraszczak, and R. Morris. Smarter cities and their innovation challenges. *Computer*, 44(6):32–39, 2011.

[NDMC+14] P. Neirotti, A. De Marco, A. C. Cagliano, G. Mangano, and F. Scorrano. Current trends in smart city initiatives: Some stylised facts. *Cities*, 38:25–36, 2014.

[NQATN18] Qassim Nasir, Ilham A Qasse, Manar Abu Talib, and Ali Bou Nassif. Performance analysis of hyperledger fabric platforms. *Security and Communication Networks*, 2018, 2018.

[NRP+21] Samudaya Nanayakkara, M.N.N. Rodrigo, Srinath Perera, G.T. Weerasuriya, and Amer A. Hijazi. A methodology for selection of a blockchain platform to develop an enterprise system. *Journal of Industrial Information Integration*, 23:100215, 2021.

[OAE18] Hashem Omrani, Arash Alizadeh, and Ali Emrouznejad. Finding the optimal combination of power plants alternatives: A multi response taguchi-neural network using topsis and fuzzy best-worst method. *Journal of Cleaner Production*, 203:210–223, 2018.

[OCF+19] Marcela T Oliveira, Gabriel R Carrara, Natalia C Fernandes, Célio VN Albuquerque, Ricardo C Carrano, Dianne SV Medeiros, and Diogo MF Mattos. Towards a performance evaluation of private blockchain frameworks using a realistic workload. In *2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, pages 180–187. IEEE, 2019.

[OEV+15] Mete Ozay, Inaki Esnaola, Fatos Tunay Yarman Vural, Sanjeev R Kulkarni, and H Vincent Poor. Machine learning methods for attack detection in the smart grid. *IEEE transactions on neural networks and learning systems*, 27(8):1773–1786, 2015.

[ÓVCB+18] María Óskarsdóttir, Tine Van Calster, Bart Baesens, Wilfried Lemahieu, and Jan Vanthienen. Time series for early churn detection: Using similarity based classification for dynamic networks. *Expert Systems with Applications*, 106:55–65, 2018.

[PC19] Jaime D Pinzón and D Graciela Colomé. Real-time multi-state classification of short-term voltage stability based on multivariate time series machine learning. *International Journal of Electrical Power & Energy Systems*, 108:402–414, 2019.

[PGF21] Remigijus Paulavičius, Saulius Grigaitis, and Ernestas Filatovas. A systematic review and empirical analysis of blockchain simulators. *IEEE access*, 9:38010–38028, 2021.

[PGK+21] Julien Polge, Sankalp Ghatpande, Sylvain Kubler, Jérémy Robert, and Yves Le Traon. Blockperf: A hybrid blockchain emulator/simulator framework. *IEEE Access*, 9:107858–107872, 2021.

[PNVNTN22] H Phan Nguyen, N Vu Ngo, and C Tam Nguyen. Study on multi-objects optimization in edm with nickel coated electrode

using taguchi-ahp-topsis. *International Journal of Engineering*, 35(2):276–282, 2022.

[PPL16] Maxime Puys, Marie-Laure Potet, and Pascal Lafourcade. Formal analysis of security properties on the opc-ua scada protocol. In *International Conference on Computer Safety, Reliability, and Security*, pages 67–75. Springer, 2016.

[PRL21] Julien Polge, Jérémy Robert, and Yves Le Traon. Permissioned blockchain frameworks in the industry: A comparison. *ICT Express*, 7(2):229–233, 2021.

[PRLT19] Julien Polge, Jeremy Robert, and Yves Le Traon. Assessing the impact of attacks on opc-ua applications in the industry 4.0 era. In *2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pages 1–6. IEEE, 2019.

[PRLT20a] J. Polge, J. Robert, and Y. Le Traon. Permissioned blockchain frameworks in the industry: A comparison. *ICT Express*, 2020.

[PRLT20b] Julien Polge, Jérémy Robert, and Yves Le Traon. A case driven study of the use of time series classification for flexibility in industry 4.0. *Sensors*, 20(24), 2020.

[PST17] S. Pongnumkul, C. Siripanpornchana, and S. Thajchayapong. Performance analysis of private blockchain platforms in varying workloads. In *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–6, July 2017.

[PTB+15] Jorge Posada, Carlos Toro, Iñigo Barandiaran, David Oyarzun, Didier Stricker, Raffaele de Amicis, Eduardo B Pinto, Peter Eisert, Jürgen Döllner, and Ivan Vallarino. Visual computing as a key enabling technology for industrie 4.0 and industrial internet. *IEEE computer graphics and applications*, 35(2):26–40, 2015.

[quo] Quorum whitepaper. `https://github.com/jpmorganchase/quorum/blob/master/docs`. Last accessed 31/03/2020.

[RD17] Sara Rouhani and Ralph Deters. Performance analysis of ethereum transactions in private blockchain. In *2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, pages 70–74. IEEE, 2017.

[RK19] Marc Rußwurm and Marco Körner. Self-attention for raw optical satellite time series classification. *arXiv preprint arXiv:1910.10536*, 2019.

[RMK16] Vasja Roblek, Maja Meško, and Alojz Krapež. A complex view of industry 4.0. *Sage Open*, 6(2):2158244016653987, 2016.

[RPRKVK19] AVS Ram Prasad, Koona Ramji, Murahari Kolli, and G Vamsi Krishna. Multi-response optimization of machining process parameters for wire electrical discharge machining of lead-induced ti-6al-4v alloy using ahp–topsis method. *Journal of Advanced Manufacturing Systems*, 18(02):213–236, 2019.

[Saa08] Thomas L Saaty. Decision making with the analytic hierarchy process. *International journal of services sciences*, 1(1):83–98, 2008.

[Sch15] Patrick Schäfer. The boss is concerned with time series classification in the presence of noise. *Data Mining and Knowledge Discovery*, 29(6):1505–1530, 2015.

[SFL17] Eugene Y Song, Gerald J FitzPatrick, and Kang B Lee. Smart sensors and standard-based interoperability in smart grids. *IEEE sensors journal*, 17(23):7723–7730, 2017.

[SH12] Patrick Schäfer and Mikael Högqvist. Sfa: a symbolic fourier approximation and index for similarity search in high dimensional datasets. In *Proceedings of the 15th International Conference on Extending Database Technology*, pages 516–527. ACM, 2012.

[SL17] Patrick Schäfer and Ulf Leser. Fast and accurate time series classification with weasel. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 637–646. ACM, 2017.

[SLdSBMQPS18] Wesllen Sousa Lima, Hendrio de Souza Bragança, Kevin Montero Quispe, and Eduardo Pereira Souto. Human activity recognition based on symbolic representation algorithms for inertial sensors. *Sensors*, 18(11):4045, 2018.

[SM13] Pavel Senin and Sergey Malinchik. Sax-vsm: Interpretable time series classification using sax and vector space model. In *2013 IEEE 13th international conference on data mining*, pages 1175–1180. IEEE, 2013.

[SOK+20] S. Smetanin, A. Ometov, M. Komarov, P. Masek, and Y. Koucheryavy. Blockchain evaluation approaches: State-of-the-art and future perspective. *Sensors*, 20(12):3358, 2020.

[SSAD19] Ankur Sharma, Felix Martin Schuhknecht, Divya Agrawal, and Jens Dittrich. Blurring the lines between blockchains and database systems: The case of hyperledger fabric. In *Proceedings of the 2019 International Conference on Management of Data*, SIGMOD '19, pages 105–122, New York, NY, USA, 2019. Association for Computing Machinery.

[SVGCPM17] César Soto-Valero, Mabel González-Castellanos, and Irvin Pérez-Morales. A predictive model for analysing the starting pitchersŠ performance using time series classification methods. *International Journal of Performance Analysis in Sport*, 17(4):492–509, 2017.

[SWTR18] H. Sukhwani, N. Wang, K. S. Trivedi, and A. Rindos. Performance modeling of hyperledger fabric (permissioned blockchain network). In *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*, pages 1–8, Nov 2018.

[SZJ17] Lyubomir Stoykov, Kaiwen Zhang, and Hans-Arno Jacobsen. Vibes: fast blockchain simulations for large-scale peer-to-peer networks. In *Proceedings of the 18th ACM/IFIP/USENIX Middleware Conference: Posters and Demos*, pages 19–20, 2017.

[TAM16] Sapna Tyagi, Amit Agarwal, and Piyush Maheshwari. A conceptual framework for iot-based healthcare system using cloud computing. In *2016 6th International Conference-Cloud System and Big Data Engineering (Confluence)*, pages 503–507. IEEE, 2016.

[THM11] Madjid Tavana and Adel Hatami-Marbini. A group ahp-topsis framework for human spaceflight mission planning at nasa. *Expert Systems with Applications*, 38(11):13588–13603, 2011.

[TKK14] Mohit Tyagi, Pradeep Kumar, and Dinesh Kumar. A hybrid approach using ahp-topsis for analyzing e-scm performance. *Procedia Engineering*, 97:2195–2203, 2014.

[TMOZ20] K. Toyoda, K. Machi, Y. Ohtake, and A. N. Zhang. Function-level bottleneck analysis of private proof-of-authority ethereum blockchain. *IEEE Access*, 8:141611–141621, 2020.

[TNV18] Parth Thakkar, Senthil Nathan, and Balaji Viswanathan. Performance benchmarking and optimizing hyperledger fabric blockchain platform. In *2018 IEEE 26th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pages 264–276. IEEE, 2018.

[TQLK18] Fei Tao, Qinglin Qi, Ang Liu, and Andrew Kusiak. Data-driven smart manufacturing. *Journal of Manufacturing Systems*, 48:157–169, 2018.

[TSD19] Huimin Tang, Yong Shi, and Peiwu Dong. Public blockchain evaluation using entropy and topsis. *Expert Systems with Applications*, 117:204–210, 2019.

[TWL+20] W.-T. Tsai, R. Wang, S. Liu, E. Deng, and D. Yang. Compass: A data-driven blockchain evaluation framwework. In *IEEE International Conference on Service Oriented Systems Engineering*, pages 17–30, 2020.

[UG15] Dejana Ugrenovic and Gordana Gardasevic. Coap protocol for web-based monitoring in iot healthcare applications. In *2015 23rd Telecommunications Forum Telfor (TELFOR)*, pages 79–82. IEEE, 2015.

[VBLS20] Gustavo Vieira, José Barbosa, Paulo Leitão, and Lucas Sakurada. Low-cost industrial controller based on the raspberry pi platform. In *2020 IEEE International Conference on Industrial Technology (ICIT)*, pages 292–297. IEEE, 2020.

[VHH16] Birgit Vogel-Heuser and Dieter Hess. Guest editorial industry 4.0–prerequisites and visions. *IEEE Transactions on Automation Science and Engineering*, 13(2):411–413, 2016.

[vV17] Jay Thoden van Velzen. Securing the insecurable? *Datenschutz und Datensicherheit-DuD*, 41(10):613–616, 2017.

[WCY+18] Bozhi Wang, Shiping Chen, Lina Yao, Bin Liu, Xiwei Xu, and Liming Zhu. A simulation approach for studying behavior and quality of blockchain networks. In *International Conference on Blockchain*, pages 18–31. Springer, 2018.

[WHH+19] Wenbo Wang, Dinh Thai Hoang, Peizhao Hu, Zehui Xiong, Dusit Niyato, Ping Wang, Yonggang Wen, and Dong In Kim.

A survey on consensus mechanisms and mining strategy management in blockchain networks. *IEEE Access*, 7:22328–22370, 2019.

[WLPB18] Xing Wang, Jessica Lin, Nital Patel, and Martin Braun. Exact variable-length anomaly detection algorithm for univariate and multivariate time series. *Data Mining and Knowledge Discovery*, 32(6):1806–1844, 2018.

[XLL+19] Xiwei Xu, Qinghua Lu, Yue Liu, Liming Zhu, Haonan Yao, and Athanasios V Vasilakos. Designing blockchain-based applications a case study for imported product traceability. *Future Generation Computer Systems*, 92:399–406, 2019.

[Xu21] J. Xu. A simulator for heavy-duty smart contracts in blockchain. In *Creative Components*, page 827, 2021.

[YK09] Lexiang Ye and Eamonn Keogh. Time series shapelets: a new primitive for data mining. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 947–956. ACM, 2009.

[YKBT19] Hui Yang, Soundar Kumara, Satish TS Bukkapatnam, and Fugee Tsung. The internet of things for smart manufacturing: A review. *IISE Transactions*, 51(11):1190–1216, 2019.

[YY10] Miao Yun and Bu Yuxin. Research on the architecture and key technology of internet of things (iot) applied on smart grid. In *2010 International Conference on Advances in Energy Engineering*, pages 69–72. IEEE, 2010.

[ZAR21] Shakeel Zafar, Zareen Alamgir, and MH Rehman. An effective blockchain evaluation system based on entropy-critic weight method and mcdm techniques. *Peer-to-Peer Networking and Applications*, 14(5):3110–3123, 2021.

[ZHLL19] L. Zheng, X. Helu, M. Li, and H. Lu. Automatic discovery mechanism of blockchain nodes based on the kademlia algorithm. In *International Conference on Artificial Intelligence and Security*, pages 605–616. Springer, 2019.

[ZHR+18] MA Zaidan, V Haapasilta, R Relan, H Junninen, PP Aalto, M Kulmala, L Laurson, and AS Foster. Predicting atmospheric particle formation days by bayesian classification of the time

series features. *Tellus B: Chemical and Physical Meteorology*, 70(1):1–10, 2018.

[ZWYG18] Jianjing Zhang, Peng Wang, Ruqiang Yan, and Robert X Gao. Long short-term memory for machine remaining life prediction. *Journal of manufacturing systems*, 48:78–86, 2018.

[ZXD+18] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang. Blockchain challenges and opportunities: A survey. *International Journal of Web and Grid Services*, 14(4):352–375, 2018.

[ZZL+18] Peilin Zheng, Zibin Zheng, Xiapu Luo, Xiangping Chen, and Xuanzhe Liu. A detailed and real-time performance monitoring framework for blockchain systems. In *Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice*, ICSE-SEIP '18, pages 134–143, New York, NY, USA, 2018. Association for Computing Machinery.