Mondragon Unibertsitatea

DimanD

DOCTORAL THESIS

Methodology and Toolset for Testing Highly Reconfigurable Programmable Logic Controllers

MIRIAM UGARTE QUEREJETA

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 814078

Thesis Directors:
Miren Illarramendi Rezabal
Goiuria Sagardui Mendieta
Iñigo Bediaga Escudero

Thesis directed to obtain the title of
Doctor in Applied Engineering
by Mondragon Unibertsitatea

Computer and Electronics Department
Faculty of Engineering
Mondragon Unibertsitatea
Arrasate, May 2023

"You cannot get through a single day without having an impact on the world around you. What you do makes a difference, and you have to decide what kind of difference you want to make"
Jane Goodall

Amona, Ama eta Aita.
Familia eta kuadrilla

Eskerrak bihotz-bihotzez beti hor egoteagatik
Miriam

Statement of Originality

I, Miriam Ugarte Querejeta, declare that this thesis is the result of my personal work, and that it has not been previously submitted to obtain another degree or professional qualification. The ideas, formulations, images, illustrations taken from other sources have been duly cited and referenced.

Miriam Ugarte Querejeta
Arrasate, June 2023

Acknowledgments

This has been a long but at the same time short journey. It has not been easy doing a PhD in the middle of a pandemic, but I have finally reached the last page of my writing. This work is the result of all the people who joined me during this journey, to which I am extremely grateful.

I would like to express my deepest gratitude to my supervisors Miren Illarramendi, Leire Etxeberria, Goiuria Sagardui and Iñigo Bediaga. Leire, I am truly grateful to have had your support at the beginning and end. Your constructive feedback, guidance and positive attitude have been key in this process. Miren, despite joining the journey during a difficult time, you managed to guide me on the right path, and your encouragement gave me the boost I needed to keep going. Thank you so much for your guidance, trust, and support, which have been vital in helping me achieve this significant milestone. Eskerrik asko benetan.

I feel privileged to be part of a fantastic team – HAZI-ISI – which made me feel at home from day one. I would like to especially thank Aitor Arrieta for generously dedicating your time, effort, and commitment to helping me. I sincerely appreciate your feedback and rigorous review in improving the quality of our article. I would also like to acknowledge Pablo Valle for offering his valuable time to assist me.

Sharon, I truly appreciate your great efforts in refining my work. Thank you for your unconditional support, guidance, and patience over these three years, particularly within the last few months.

My colleagues at IDEKO. I am fortunate to have had such an outstanding mentor as Gorka. You embody the essence of commitment. Working with you has been a great experience, which brought a sense of fun and excitement into my work, keeping me up and motivated. Jon Peñalper, for assisting me with the Test Suite environment, and Jose Luis for dedicating so much time and effort to helping me. Your professionalism and commitment are truly appreciated. You all made my stay at IDEKO a joyful and fruitful experience.

Thanks to my colleagues at the University of Nottingham – Jack, Svetan, Joe H, Joe G, Joel, Giovanna, Dave S, Dave B, Fan, Agajan, Hamood, Karina, Sara, Karol, Basem... – for making the secondment a fruitful experience, and many thanks Helena, Kryssa and Nancy for organising everything. I would also like to thank my colleagues at KTH – Antonio Maffei, Fabio Monetti and Eleonora Boffa – for welcoming me warmly, the days seemed brighter than they were.

My DiManD family – Hien, Joaquin, Angela, Agajan, Fan, Hamood, Jose, Trunal, Luis, Terrin, Nathaly, Fabio – I hope to keep in touch with you. Thank you Miren and

Felix for your great commitment to managing this project.

My colleagues of Galarreta, Galarreta PhD students, and former PhD students for very warmly hosting me at your faculty.

Mondragon University and DiManD for giving me the opportunity to work on this thesis.

Eskerrik asko nere kuadrilla, Londoneko kuadrilla, my Inmarsat family, eta alboan egon zareten gainontzeko lagun guztiak, arnasgune izan baitzarete. Zuekin hau eta beste horrenbeste momentu ospatzeko desiatzen.

Azkenik, nere familia osoaren babesa eskertu nahiko nuke. Eskerrik asko bihotz-bihotzez Amona, Ama, Aita, "Andoni", eta nola ez, Xenek, beti pozez irrikan agurtzeko ate ondoan egotegatik.

Abstract

The manufacturing industry has entered a new era of highly changeable customer requirements and tailor-made products. To meet market demands, manufacturing systems need to be redefined to be able to cost-effectively produce a wide variety of products. Sustainability is key in this endeavour, maximising the utilisation of all available resources to address new bills of processes. To achieve this, however, the development of new capabilities are imperative to seamlessly adapt the manufacturing sector to the new paradigm.

At the core of the production system, the Programmable Logic Controller (PLC) orchestrates all necessary resources to initiate new processes, including modifications to the PLC program. For this reason, PLC software must be thoroughly tested after each adaptation, to ensure operational safety. At present, testing is conducted during commissioning, which typically takes place at the end of the development process, resulting in a significant impact in time and cost. In addition, the manual nature of testing practices requires considerable effort, and leaves the PLC open to errors. All these issues are further compounded in frequently changing and adaptive environments, hence the need to adopt new practices.

In this research project, therefore, we present an automated and cost-effective methodology to test highly reconfigurable PLC programs in industry. The presented approach is industry oriented, and thus we focus on the transfer of best-practices in software engineering, to the manufacturing environment. To this end, we introduce a theoretical virtual commissioning framework to enhance commissioning practices. Moreover, our approach is based on testing the logic of Functional Block Diagram (FBD) programs, one of the most widely employed PLC programming languages in Europe. Our methodology comprises the generation of the test cases based on the IEC 61131-3 standard, cost-effective test selection of the test cases, and simulation-based testing in commercially used automation solutions (the Siemens TIA portal) with the use of test oracles.

The proposal is validated with two real industrial case studies: 1) Omnifactory, which is the future automated aerospace assembly demonstrator, recently launched at the University of Nottingham, and 2) the machine tool manufacturing industry at Danobatgroup. The results demonstrate that by automating conventional practices our methodology can effectively test real industrial PLC programs, which ultimately reduces costs and time, and ensures the reuse of available resources.

Laburpena

Manufaktura-industria aro berri batean murgiltzen ari da, bezeroen eskari neurrigabeak eta neurrira egindako produktuen beharrak bultzatuta. Merkatuaren beharrak asetzeko asmotan, errentagarriak diren fabrikazio-sistemak berriro definitu beharrean aurkitzen gara. Jasangarritasuna kontutan hartzea funtsezkoa da prozesu honetan, eskura dauden baliabideak berrerabiliz prozesu berriak burutu ahal izateko. Halaber, era berri honi aurre egiteko digitalizazioak bultzatutako teknologia eta ezagutza berriak baliatu behar ditugu.

Kontrolatzaile logiko programagarriak (PLC-ak) industria automatizazioan erabiltzen diren gailu elektronikoak izanik, egoera berriei erantzuteko bere baitan dauden baliabide guztiak uneoro kudeatu behar dituzte. Prozesu honek aldiz, PLC-aren softwarea aldatzea dakar. Hori dela eta, sistemak behar bezala funtzionatzen duela bermatzea ezinbestekoa da, hots, aldaketen ostean akatsik egon ez dela egiaztatzea. Tradizionalki hau eskuz egin ohi den jarduera da, ahalegin haundiak eskatzen dituelarik. Honetaz gain, sistemaren funtzionamendu egokia ez da azken momentura arte begiratzen, eta ondorioz, ezusteko akatsak aurkitu izan ohi dira. Uneoroko aldaketak eskatzen dituen era berri honek, arazo guzti hauek are gehiago larrituko lituzke.

Tesi honek PLC-en software-ari zuzendutako test jarduera automatizatu eta eraginkor bat aurkeztea du helburu. Gauzak honela, Europa mailan gehien erabiltzen den PLC lengoaietako batetan oinarritzen gara, zehazki FBD lengoaian. Lehenik eta behin, biki digitaletan oinarritutako esparru teoriko bat proposatzen da, jarduera hauek modu eraginkor batetan aplikatu daitezen. Garatutako metodologiak hiru urrats nagusi ditu: 1) IEC 61131-3 estandarrean oinarritutako test instantziak sortzea, 2) test hauen optimizazioa, eraginkorrenak soilik aukeratze aldera, eta 3) test jarduerak merkatuan aurkitzen diren simulazio testuinguruetan exekutatzea, horretarako orakulu testak erabiliz.

Industriara bideratuko tesia izanik, industrian gehien erabiltzen den automatizazio soluzioetako batetan inplementatu da, hain zuzen Siemens-eko TIA Portal produktuan. Honekin batera, software ingeniaritzako jarduerak eta jakintza industriara eramatea lortu da. Metodologia balioztatzeko, industriako bi kasu erreal erabili dira: 1) Omnifactory, Nottinghameko Unibertsitatean garatutako etorkizuneko aeroespazioko muntaketara bideratutako automatizazio plataforma, eta 2) makina-erreminten fabrikazio-industrian Danobatgroup-eko makinak kudeatzen dituen PLC-a. Emaitzen arabera, aurkeztutako metodologia test jarduerak modu eraginkorrean automatizatzeko gai da, azken finean, kostuak eta denbora murriztuz, eta baliabideen berrerabiltasuna uztartuz.

Resumen

La industria manufacturera se está embarcando en una nueva era, impulsada por una demanda sin precedentes y una necesidad de productos hechos a medida. De este modo, los sistemas de fabricación deben redefinirse para poder producir una gran variedad de productos de manera rentable. La sustentabilidad es clave en este proceso, prestando especial atención a la reutilización de los recursos disponibles para abordar nuevos procesos. Sin embargo, para adaptar la industria al nuevo paradigma exitosamente, es necesario el desarrollo de nuevas capacidades impulsados por la digitalización.

El controlador lógico programable (PLC) es la unidad central del sistema de producción, y debe de orquestar todos los recursos necesarios para abordar nuevos procesos. Esto implica modificaciones en el programa del PLC. Por lo tanto, el software del PLC requiere un proceso de verificación y validación exhaustivo para garantizar la correcta funcionalidad del sistema. Tradicionalmente, este proceso se lleva a cabo durante la puesta en marcha, donde se testea todo el sistema por primera vez. En general, los sistemas son manualmente testeados durante este proceso, el cual requiere un esfuerzo significativo y que en gran parte esta sujeto a errores. Todos estos problemas se agravan aún más en entornos donde los cambios son frecuentes, y por lo tanto existe la necesidad de adoptar nuevas prácticas.

En esta tesis se presenta una metodología automatizada para testear PLCs altamente reconfigurables de una manera eficaz y efectiva en la industria. Primeramente, se ha propuesto un marco teórico donde los gemelos digitales son clave para mejorar las prácticas de la puesta en marcha. De este modo, el trabajo principal de esta tesis se ha centrado en testear la lógica de los programas FBD, uno de los lenguajes de programación de PLC más utilizados en Europa. La metodología cosiste principalmente de tres pasos, 1) generación de los tests basados en el estándar IEC 61131-3, 2) optimización de los tests para seleccionar los más eficaces y efectivos, y 3) testeo de PLCs basado en entornos de simulación comerciales mediante el uso de oráculos.

Esta tesis está orientado a la industria y, por lo tanto, la solución expuesta se ha implementado en el entorno TIA Portal de Siemens, transfiriendo las buenas prácticas de ingeniería de software a la industria. De este modo, la propuesta se ha validado con dos casos de estudio industriales reales 1) Omnifactory, el futuro demostrador de ensamblaje aeroespacial automatizado desarrollado en la Universidad de Nottingham, y 2) Industria de fabricación de máquina-herramienta en Danobatgroup. Según los resultados, la metodología presentada es válida para automatizar el testeo de programas PLC industriales eficazmente, lo que en definitiva reduce costes y tiempo, y permite la reutilización de los recursos.

Contents

List of Figures

List of Tables

Acronyms

ABR	Arithmetic Block Replacement
AMS	Autonomous Manufacturing Systems
AND	Logical conjunction operator
AWL	Assembly Language for STEP 7
BC	Basic Coverage
BP	Bistable Processor
C/DC	Condition/Decision Coverage
CC	Condition Coverage
CCC	Complex Condition Coverage
CNC	Computer Numerical Controls
CPS	Cyber-Physical Systems
CPU	Central Processing Unit
CTU	Count Up operator
DB	Data Block of a Function Block Diagram
DC	Decision Coverage
DiManD	Digital Manufacturing and Design
DMS	Dedicated Manufacturing Systems
DPC	Data Path Condition
DSR	Design Science Research
EAS	Evolvable Assembly Systems
ERP	Enterprise Resource Planning
ESR	Early State Researcher
ET	Execution Time

FA3D	Future Automated Aerospace Assembly Demonstrator
FAT	Factory Acceptance Testing
FB	Function Block of a Function Block Diagram
FBD	Function Block Diagram
FC	Function Call of a Function Block Diagram
FDC	Fault Detection Capability
FIO	Feedback loop Insertion Operator
FMS	Flexible Manufacturing Systems
GA	Genetic Algorithm
GE	Greater than or Equal to operator
HiL	Hardware-in-the-Loop
HV	Hypervolume indicator
ICC	Input Condition Coverage
ICT	Information and Communications Technology
IL	Instruction List
IoT	Internet of Things
ITN	Innovation and Training Network
JCR	Journal Citation Report
KAIST	Korea Advanced Institute of Science and Technology
KNICS	Korea Nuclear Instrumentation and Control System
LD	Ladder Diagram
LDO	Logical Block Deletion Operator
LIO	Logical Block Insertion Operator
LRO-I	Logical Block Replacement Operator-Improved
MC/DC	Modified Condition/Decision Coverage
MCC	Multiple Condition Coverage
MCSAT	Model Construction Satisfiability algorithm
MES	Manufacturing Execution System
NSGA-II	Non-dominated Sorting Genetic Algorithm II
OB	Organisational Block of a Function Block Diagram

OR	Logical disjunction operator
PLC	Programmable Logic Controller
PLM	Product Life-cycle Management
POU	Program Organisation Unit
PT	Pulse Time
PTC	Propagation Toggle Coverage
RiL	Reality-in-the-Loop
RMS	Reconfigurable Manufacturing Systems
ROI	Return on Investment
RQ	Research Question
RS	Random Search
RUL	Remaining Useful Life
SAT	Site Acceptance Testing
SBST	Search-based Software Testing
SC	Statement Coverage
SCADA	Supervisory Control and Data Acquisition
SCIE	Spanish Informatics Scientific Society
SCL	Structured Control Language
SiL	Software-in-the-Loop
SMT	Satisfiability Modulo Theories
SMV	Symbolic Model Verifier
SoS	System of Systems
SR	Set-Reset
ST	Structured Text
TET	Total Execution Time
TON	Timer-on-Delay
TR	Test Requirement
TRL	Technology Readiness Level
UML	Unified Modelling Language
UNOTT	University of Nottingham

VRO	Variable Replacement Operator
VRO-I	Value Replacement Operator-Improved
WOS	Web of Science
XOR	Logical exclusive disjunction operator

Part I

Foundation and Context

Chapter 1

Introduction

Contents

This chapter introduces the motivation and scope of this research work, as well as the issues addressed during the course of the project. The main technical contributions are summarised in Section 1.3, and published research and journal articles are detailed in Section 1.4. Lastly, additional research activities that were carried out are presented in Section 1.5.

1.1 Motivation and Scope of the Research

This thesis is part of the Digital Manufacturing and Design (DiManD) Innovation and Training Network (ITN) programme, which is a high-quality multidisciplinary, multi-professional, and cross-sectorial European research and training network focused on Industry 4.0. The main objective is to foster industrial competitiveness and sustainability in the European manufacturing sector, by leveraging a technology-driven infrastructure that can produce a wide range of dependable products in a cost-effective manner. In effect, this program promotes a new way of manufacturing high-quality goods whilst complying with sustainable development goals throughout the whole life cycle of the product.

Present-day manufacturing is undergoing a shift from the traditional mass production-based paradigm to offering customised products. To respond to ever-changing customer needs, production systems must be highly reconfigurable and adaptive. Instead of solely focusing on specific tasks, these systems need to deliver a wider variety of products with rapid turn-around. A critical attribute to succeed in this transition is agility.

Adaptive manufacturing systems require re-configuring the existing layout, adding new elements, and modifying or removing existing ones based on the needs of the moment. Most of these reconfigurations involve changes in the manufacturing process, and orchestration is thus a complex and crucial task. PLCs are commonly used to control industrial manufacturing processes, and these must be adapted to orchestrate new processes in the manufacturing system. Every change in the PLC code must be fully tested, however, to ensure there are no errors before being deployed into operations. At present PLC testing is mostly conducted manually. This is a tedious task which demands considerable time and effort, and is highly error-prone. The problem is further compounded in reconfigurable and adaptive production systems, as these continuously undergo changes which require rigorous testing before implementation.

Cyber-Physical Systems (CPS), which include PLCs, are traditionally tested during the commissioning phase of the development process. This is not ideal, as commissioning occurs at the very end of the chain-based manufacturing process, in which each step depends on the previous one (design -> planning -> engineering -> assembly -> commissioning). Despite the evident logic of this structure, any unexpected issue in any of these steps can result in serious delays in downstream processes. Hence, commissioning at the end of the development process can incur significant costs, as the system is not tested until everything is in place.

The advent of Industry 4.0 has placed virtual commissioning centre stage, as a series of verification tasks between multiple engineering disciplines can be instigated early in the development process. This has obvious benefits in that it can significantly reduce commissioning time, errors, and costs. This potential remains under-exploited in industry, however, as virtual commissioning is still rarely applied until the latter stages of the manufacturing chain.

In the manufacturing sector companies are reluctant to invest in simulation technologies for virtual commissioning [UQEESM+21, LP14, KWL+20]. This is particularly true of machine tool manufacturers. One of the main issues is interoperability: each vendor delivers a single solution with proprietary communication protocols, thereby limiting the integration of other products. Integrating simulation tools from different domains, as well as the numeric controller to perform virtual commissioning, only serves to exacerbate the problem. Moreover, virtual models may lack accuracy, as they do not reflect the conditions of physical devices, and require additional validation to be performed through conventional commissioning procedures. As a result, companies are unable to justify the Return on Investment (ROI), which poses a serious obstacle to the adoption of virtual solution strategies.

Digital twins are thus well positioned to enhance virtual commissioning practices as they can address synchronisation issues and misalignments between virtual representations and the physical system. In addition, while virtual commissioning typically consists of various simulation technologies commissioned in silos by distinct engineering disciplines, the digital twin can integrate all these models into a single framework. However, no commercial holistic and multi-domain solution technologies are currently available.

Agile testing methodologies which exploit the virtual commissioning framework are critical to maintaining a competitive edge in European manufacturing. These include methods to detect defects, malfunctioning, and potential errors in the design. These can save time, resources, and overall costs by preventing corrections in the commissioning stage.

To overcome the gap in integrating software testing techniques in industrial environments, the following challenges need to be first addressed:

- **Ch1:** dealing with turbulent and competitive markets.

- **Ch2:** fostering sustainable goals.
- **Ch3:** interoperability issues due to vendor proprietary protocols and languages.
- **Ch4:** time to market and costs as a result of commissioning at latter stages of the development process.
- **Ch5:** lack of collaborative and holistic solutions for virtual commissioning, in which different multi-domain disciplines can work collaboratively.
- **Ch6:** difficulty in quantifying the ROI.
- **Ch7:** time and effort required to test and commission PLCs, which are further increased in adaptive manufacturing systems.
- **Ch8:** dealing with frequent errors as manual testing is error-prone.
- **Ch9:** lack of knowledge transfer to industrial environments.

In light of the issues identified above, this research project focuses on providing an agile mechanism to test industrial PLC programs based on software engineering practices (*Ch9*). To accomplish this, we propose a theoretical virtual commissioning framework (*Ch3, Ch5*) to foster the continuous integration of PLC software (*Ch7*). Our work is aligned with the objectives of DiManD, and the methodology we present provides an automated (*Ch8*) cost-effective (*Ch6, Ch7*) approach for testing PLC programs that are subject to frequent reconfigurations (*Ch1*). Our approach maximises the use of existing manufacturing resources (*Ch2*) and delivers high reliability while reducing commissioning time (*Ch4*).

1.2 Research Methodology

The research methodology follows the guidelines of the Design Science Research (DSR) framework presented by Vaishnavi et al. [VK15]. The methodology comprises a five steps process, as indicated in Figure 1.1.

- **Awareness of problem**: the first step is the acknowledgements of a problem. In this case, the initial problem could be used to tackle a research question. The output might be a formal or informal proposal. To this end, with the aim of increasing awareness and obtaining a comprehensive understanding of the problem, an extensive literature review was conducted (Chapter 3). In addition, we supplemented our findings with an industrial survey to obtain further insights regarding the research

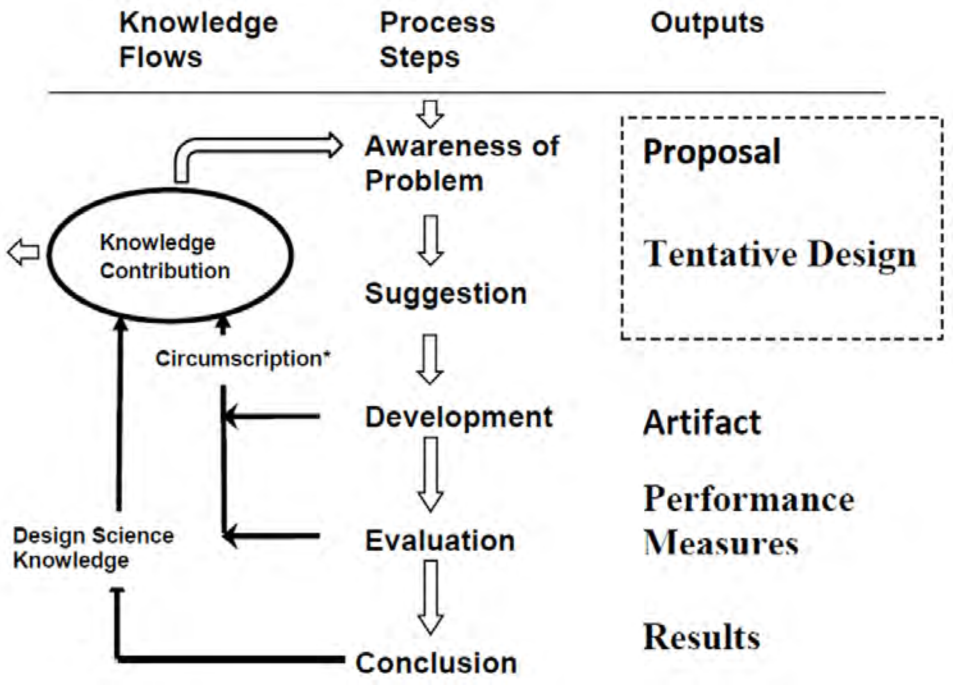


Figure 1.1: General overview of the research methodology [VK15]

questions we posed about virtual commissioning practices in the manufacturing industry (Chapter 4).

- **Suggestion**: in this step, a tentative solution is suggested to address the initial problem. The suggestion might come from the existing knowledge, and is usually an integral part of a formal proposal. In response to the identified gaps, we proposed a theoretical framework and suggested 4 hypotheses (Chapter 5), including a hypothesis related to the theoretical framework, and three technical.

- **Development**: development and implementation of the suggested tentative solution. As a result, a novel artefact is created. Specifically, we developed three technical artefacts to address the technical hypothesis raised in the previous step. These artifacts are described in Chapter 6, Chapter 7 and Chapter 8.

- **Evaluation**: in this step, the partially or fully implemented artefact is assessed based on certain criteria. This phase involves an analytical subphase, where the hypotheses are tested, and the resulting knowledge is looped back to the previous stages. To this end, we carried out an empirical analysis to evaluate each of the technical artefacts and thus, respond to the hypotheses. These were individually evaluated in the corresponding chapter.

■ **Conclusion**: the research cycle finishes when the resulting artefact complies with the requirements specified in the previous phases. The knowledge gained within this process is fed back for the next research cycle. In this case, we used all this knowledge to present a business model (Section D.2) of the developed solution, with the aim of assessing the impact of our proposal in the market.

1.3 Technical Contributions

In this thesis, we present a methodology to cost-effectively test industrial PLC programs with the aim of reducing commissioning time, while ensuring a high level of reliability. The ultimate goal is to maximise the reuse of manufacturing resources to address new bills of process, by promoting the continuous integration of PLCs. To this end, we introduce a theoretical framework for virtual commissioning. Our simulation-based approach automatically tests Siemens PLCs – one of the most widely used in industry – by applying software engineering techniques to commercially available automation solutions. The main contributions of this thesis are summarised as follows:

1. **Preliminary work:** an empirical survey to identify industrial requirements, challenges, and needs. This process involved industrial and academic stakeholders, and provided first-hand information on virtual commissioning practices. The results of the study were supported with an expanded review of the literature to pinpoint emerging trends and research gaps.

2. **Theoretical framework:** a theoretical virtual commissioning framework designed to address the identified industrial needs, which ultimately could facilitate the continuous integration of PLCs in the ever-changing manufacturing field.

3. **Technical contribution 1:** a test case generation and evaluation approach for IEC 61131-3 FBD programs. Effectively, the approach included coverage-based, mutation-based, and random test cases. This was achieved by extending the capabilities of existing tools and incorporating a new solver capable of handling a broad range of IEC 61131-3 FBD programs, including non-linear arithmetic functions. We also developed an automated test evaluation approach, in which cost-effective metrics were calculated, and test oracles were obtained from test execution results. The metrics are used to optimise test selection, whereas test oracles are employed to validate the simulation results.

4. **Technical contribution 2:** a cost-effective test selection approach. To this end, we implemented a search-based test selection algorithm to optimise the selection of

test cases. The optimisation focused on maximising fault detection capability while minimising execution time. This was accomplished by defining a fitness function based on the derived cost-effective metrics, which included coverage, execution time, and fault detection capability.

5. **Technical contribution 3:** a methodology to test Siemens PLC programs. The solution is based on a Software-in-the-Loop (SiL) approach, which uses the Siemens automation platform – TIA Portal – to perform the tests against the Siemens virtual controller. Test oracles were used to assert the results of the simulation. The methodology comprised six steps, including 1) PLC code standardisation, 2) test case generation, 3) test case evaluation, 4) test case selection, 5) TIA Portal application test generation, and 6) simulation-based application test execution.

1.4 Publications

In total, five peer-reviewed publications were published in journals and at conferences during the course of this project. At the time of writing, a further two articles were under review, and one final article was ready for submission.

The publications are scored according to the Journal Citation Report (JCR) from Web of Science, CiteScore from Scopus, and Scimago Journal Rank (SJR) from Scimago. In addition, the conference publication papers related to software engineering are ranked by the ranking systems supported by the Spanish Informatics Scientific Society (SCIE (www.scie.es)).[1]

1.4.1 Journal Articles

During the course of this dissertation, two journal articles were published:

- Miriam Ugarte Querejeta, Leire Etxeberria, Goiuria Sagardui, Gorka Unamuno, and Iñigo Bediaga. “Virtual commissioning in machine tool manufacturing: a survey from industry” in DYNA Ingeniería e Industria (2021), 96 (6), pp. 612-619. **JCR:** 2.070 Engineeering Multidisciplinary Q3 51/92. **SJR:** 0.160 Engineering C4 333/414 2021 2022-07-11. **CiteScore 1.3:** General Engineering CS3 178/300.

- Miriam Ugarte Querejeta, Miren Illarramendi Rezabal, Gorka Unamuno, Jose Luis Bellanco, Eneko Ugalde, and Antonio Valor Valor. “Implementation of a holistic digital twin solution for design prototyping and virtual commissioning”. IET Collaborative Intelligent Manufacturing (2022), vol. 4 (4), pp. 326-335. **SJR:**

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

0.568 Industrial and Manufacturing Engineering C2 104/368. **CiteScore:** 2.8. Industrial and Manufacturing Engineering C3 151/338.

At the time of submitting this thesis, a journal article was also in progress for later submission to the Journal of Manufacturing Systems.

- Fan Mo, Miriam Ugarte Querejeta, Joseph Hellewell, Hamood Ur Rehman, Miren Illarramendi Rezabal, Jack C. Chaplin, David Sanderson, Svetan Ratchev. "PLC Orchestration Automation to enhance Human-Machine Integration in Adaptive Manufacturing Systems". **JCR:** 9.498 Engineering, Manufacturing Q1 7/51.

In addition, as part of the work carried out in the workpackages within the DiManD network, a journal article was sent for review in the Journal of Intelligent Manufacturing, which has been accepted for publication:

- Jose Antonio Mulet Alberola, Luis Alberto Estrada-Jimenez, Hien Nguyen Ngoc, Trunal Patil, Miriam Ugarte Querejeta, Angela Carrera-Rivera, Mauro Onori and Antonio Maffei. "Towards autonomous manufacturing automation: Analysis of the requirements of self-x behaviours". **JCR:** 7.136 Engineering, Manufacturing Q1 11/51.

1.4.2 International Conferences

Two international conference articles were published at ISM and CIRP:

- Miriam Ugarte, Leire Etxeberria, Gorka Unamuno, Jose Luis Bellanco, and Eneko Ugalde. "Implementation of Digital Twin-based Virtual Commissioning in Machine Tool Manufacturing" in Procedia Computer Science (2022), 200, pp.527-536. **CiteScore:** 3.0 General Computer Science CS2 71/226.

- Itziar Ricondo, Alain Porto, Miriam Ugarte. "A digital twin framework for the simulation and optimisation of production systems" in Procedia CIRP (2021), vol. 104, pp. 762-767. **CiteScore:** 3.3 Industrial and Manufacturing Engineering CS2 107/336.

Furthermore, at the time of submitting this thesis a conference paper was sent for review by the Genetic and Evolutionary Computation Conference (GECCO 2023).

- Miriam Ugarte Querejeta, Pablo Valle, Aitor Arrieta, Eunkyoung Jee, Lingjun Liu, Miren Illarramendi Rezabal. "Cost-Effective Test Selection for Functional Block Diagram Programs". **SCIE**: A.

1.4.3 Workshops

A conference article was published at the SAFECOMP-DepDevOps workshop.

- Miriam Ugarte Querejeta, Leire Etxeberria, and Goiuria Sagardui. "Towards a DevOps Approach in Cyber Physical Production Systems Using Digital Twins" at the International Conference on Computer Safety, Reliability, and Security (2020), pp. 205–216. **SCIE**: B-.

1.5 Other Related Activities

In addition, the PhD student has contributed to other activities that furthered her training and professional development as a researcher. These activities included dissemination activities, service to the community, participation in DiManD schools, national and international secondments, and work in collaborative workpackages within the DiManD project.

1.5.1 Dissemination

The work developed during this research project, as well as the DiManD project, were disseminated in the following events:

- DiManD project dissemination within the R&D Projects Dissemination Event at the 3rd International Conference on Industry 4.0 and Smart Manufacturing (ISM 2021).
- Roundtable in the STEAM Technology and Science Week at GARAIA Technology Park (2020).
- Short video recording on digital twins and Industry 4.0 for the Citizen Engagement Event (2020).
- Thesis and DiManD project presentation at MGEP as part of the *NiZuGu Mugituz* career counselling project.
- Short video interview that describes this research project within the scope of the DiManD project, published on the www.dimanditn.eu webpage.
- Short video recording that describes this research project as part of the DiManD project on the Mondragon Uniberstitatea, Faculty of Engineering communication channel (2022).

1.5.2 Service

The PhD student was involved in the following activities as service to the research community:

- Early State Researchers (ESRs) delegate on the DiManD supervisory board.
- Peer-reviewer at the 3rd International Conference on Industry 4.0 and Smart Manufacturing (ISM 2021).

1.5.3 Workpackages

Complementary research was carried out in several multidisciplinary work packages, within the framework of the DiManD programme. This contributes to an inter-sectoral and interdisciplinary collaborative framework, which facilitates cooperation, communication, and teamwork among all participants. In particular, the PhD student was involved in WP3 Tasks 3.1 and 3.2, and WP4 Tasks 4.2 and 4.3. The outcomes are published at www.dimanditn.eu.

WP3: Integration of computation, networking, and physical processes into CPS. This workpackage was led by KTH Royal Institute of Technology.

- **Task 3.1**: Identification of major challenges to the industrial adoption of CPS focusing on the integration of computation, networking and physical processes towards the vision of CPS.
- **Task 3.2**: Development of a CPS architecture that provides the necessary framework for the development of CPS.

As part of Task 3.1, a comprehensive state-of-the-art analysis was conducted within the field of autonomous CPS, Circular Manufacturing Systems, and Evolvable Production Systems. Further research was also focused on Self-X behaviours, such as self-adaptation, self-configuration, self-organisation, and self-learning to establish the requirements of the CPS architecture.

Task 3.2 involved Quality Function Deployment methods focusing on the requirements established in Task D3.1, as a mechanism to measure the requirements to form the new paradigm. The technical requirements were therefore prioritised and mapped onto the 5 CPPS architecture and MAPE-K loop to build the new paradigm of autonomous and sustainable manufacturing systems.

WP4: Autonomous, context aware manufacturing platforms. This workpackage was led by the University of Nottingham.

- **Task 4.2**: Development of a data model for proactive intelligent products.
- **Task 4.3**: Development of adaption strategies for context-aware autonomous systems.

As an outcome of Task 4.2, two data models were designed: 1) a process data model to represent manufacturing process requirements and production knowledge, and 2) a runtime condition model to represent the capabilities and status of manufacturing resources. The main objective was to develop an intelligent product that could potentially produce a new bill of process by matching these two data models.

Finally, the scope of Task 4.3 was to define an adaption strategy to reschedule tasks in the event of unforeseen situations. This was carried out based on the data models developed in Task 4.2.

1.5.4 Schools

Attending a series of school events was a key part of the DiManD project. The main objective was to receive network-wide multidisciplinary training to gain scientific knowledge and obtain complementary skills provided by academic and industrial partners of the DiManD network.

- **Welcome Event**: Introduction to the project (February 2020)
- **School event 1**: Machine learning, deep learning, soft-sensors and datathon (October 2020)
- **School event 2**: Cyber safety and security (March 2021)
- **School event 3**: Distributed agent-based control (July 2021)
- **School event 4**: Mixed reality systems (November 2021 - January 2022)
- **School event 5**: Critical design review of the integrated training and demonstration platform and workshop with industry (February 2022)
- **School event 6**: Challenge-driven research for digital manufacturing (June 2022)

1.5.5 Integrated Project

As part of the key activities of DiManD, the PhD student worked with other ESRs on a collaborative integrated project for the national demonstrator and testbed for smart manufacturing systems, namely Omnifactory, at the University of Nottingham.

The PhD student focused on designing a methodology to ensure the PLC program is correctly implemented. This is key for the Omnifactory, as the PLC will need to orchestrate new processes to address any sudden manufacturing needs. Hence, new PLC programs should be frequently tested.

The roadmap of the integrated project consisted of the following stages, which were held during the school events:

- Introduction of the integrated project and demonstration platform Omnifactory (Welcome event).
- Preliminary design review with industry (School event 1).
- Critical design review with industry (School event 5).
- Presentation of the integration of the individual projects (School event 6).

1.5.6 Secondments

Secondments are activities undertaken to gain intersectoral and international experience while collaborating with other institutions. The PhD student carried out a 4-month industrial secondment at IDEKO in Spain (national secondment), one month of stay at ULMA Embedded Solutions (UES) in Spain (national secondment), one month of stay at the University of KTH in Sweden (international secondment), and two one-month secondments at the University of Nottingham (UNOTT) in the UK (international secondment). This section briefly describes the activities and outcomes of these secondments.

IDEKO: the IDEKO Secondment was carried out in two rounds, which included an early stay in the beginning of the thesis, and a final stay to finalise the results. The main objective of the first stay was to gather industrial requirements and gain expertise in the field of manufacturing technologies for machine tools. Specifically, the industrial research context focused on digital twins for machine tools, and thus existing needs and challenges with regard to digital twins and virtual commissioning were explored. An industrial survey was carried out (see Section 4) to gather feedback from the industry. This helped gain awareness of existing industrial needs and identify gaps for further research. During the final stay, we focused on the use case implementation, in which we conducted experimental tests on their machine tool solutions.

UES: the main objective of the UES secondment was to gain knowledge in the area of test management engineering. To this end, the PhD student received a training course focused on the IBM test management platform. The training comprised the development of test plans, test cases, test scripts and test suites, as well as the management of test execution results.

KTH: this secondment focused on the study of manufacturing strategies to benchmark current manufacturing trends and lead the shift towards truly autonomous and sustainable production systems. As an outcome, the ESRs Miriam Ugarte Querejeta and Jose Antonio Mulet Alberola kicked off the deliverable under the supervision of Prof. Antonio Maffei. In particular, the integrated business models of the engineer-to-order paradigm were analysed.

UNOTT: the secondment involved two one-month stays, six months apart. The first stay was mainly focused on the development of PLC programming and testing solutions for the Omnifactory. The Omnifactory was used as a case study to test our methodology, as part of the collaborative integrated project of DiManD. During the second stay, a collaborative journal paper was written, which is intended for submission to the Journal of Manufacturing Systems.

1.6 Document Structure

The thesis is structured as follows: the first part of the thesis corresponds to the foundation and context. Chapter 1 introduces the main motivation of the thesis, the employed research methodology, the contributions, the achieved publications and the activities accomplished by the PhD student. Basic background, as well as terminology used during the rest of the document, is provided in Chapter 2. Chapter 3 gives an overview of the state of the art and highlights the most relevant studies related to this thesis. An empirical survey is presented as preliminary work in Chapter 4. The theoretical framework is explained in Chapter 5, including the research objectives, the research hypotheses, an overview of the proposed solutions and the employed case studies.

The second part focuses on test data generation for FBD programs. In Chapter 6 we provide a test generation and evaluation approach to generate mutation-based and coverage-based test cases, as well as the evaluation of the test cases to obtain cost-effective metrics and test oracles.

The third part corresponds to test optimisation. Specifically, Chapter 7 presents a multi-objective test selection methodology based on cost-effective metrics.

The fourth part corresponds to TIA Portal Testing. In Chapter 8 we present our methodology to test Siemens PLC programs with the use of the virtual controller.

To conclude, in the final remarks part in Chapter 9, we summarise the contributions of the thesis, we validate the hypotheses and we discuss the main limitations of the proposed solution. Furthermore, we propose and discuss future research.

Chapter 2

Technical Background

Contents

This research project has been conducted within the scope of **Industry 4.0**, with the objective of contributing to an autonomous and sustainable manufacturing infrastructure in Europe. Our work focuses on the development of manufacturing infrastructure that can quickly respond to turbulent market needs, by providing a fast integration and configuration of CPS resources to maximise utilisation. **Virtual commissioning** plays a key role in this project, as it ensures an agile integration environment that tests and validates new configurations in the field.

PLCs are the core unit of the production system, and often require updates to address new customer needs. Any such changes must be fully tested to ensure there are no errors in the code. To this end, our study leverages advances in **CPS testing techniques** and employs **regression testing** solutions to cost-effectively test recent modifications in the PLC code. This will help maximise existing manufacturing resources and deliver sustainability and competitiveness.

The aim of this chapter is to familiarise the reader with the aforementioned concepts. As such, Section 2.1 introduces Industry 4.0, by describing the manufacturing shift over the last years, and outlines the key technologies. In Section 2.2, we outline existing virtual commissioning practices. Lastly, Section 2.3 details Software testing practices, with a particular focus on PLC code testing.

2.1 Industry 4.0

Since the first industrial revolution, the manufacturing sector has undergone significant periods of transformation to meet the rapidly changing needs of the market (Figure 2.1). The first industrial revolution took place at the end of the 18th century, when water and steam-powered machines were first introduced to mechanised production. In 1870, electricity was harnessed as the primary source of power to meet the demands of mass production, ushering in the second great transformation. During the third industrial revolution, Information and Communications Technology (ICT) and electronics were developed to deliver automated production systems.

Today, we find ourselves in the midst of the 4th industrial revolution, referred to as Industry 4.0. The term originated in 2011 as "Industrie 4.0", introduced by the German Federal Government as a future project to build smart factories [Mac14]. Industry 4.0 brings together Cyber-Physical Systems, the Internet of Things (IoT), and Big Data technologies that require dynamic, agile, and customised manufacturing systems.

The shift from high-volume mass production to low-volume high bespoke has led to various manufacturing paradigms, including Dedicated Manufacturing Sys-

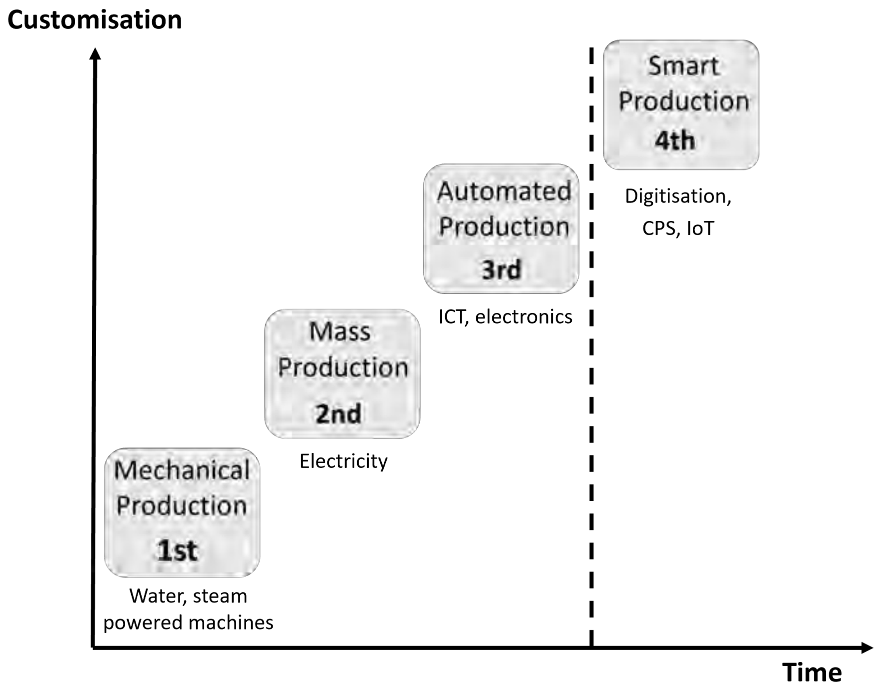


Figure 2.1: Industry evolution, from Industry 1.0 to Industry 4.0

tems (DMS), Flexible Manufacturing Systems (FMS), Reconfigurable Manufacturing Systems (RMS), and more recently, Autonomous Manufacturing Systems (AMS).

Dedicated Manufacturing Systems: DMS are based on fixed automation intended to address mass production requirements [GŚ12]. These comprise a series of fixed operating machines designed to produce a single product at maximum efficiency [SMRM20]. Such systems are extremely rigid and can be difficult to alter when the demands of the customer change [ZLGH06].

Flexible Manufacturing Systems: FMS offer higher product variability than DMS as they are composed of machines that can produce a variety of products [Ren10]. FMS are suitable for medium to small batch production that produces families of products in a flexible manner [GŚ12]. However, the range of flexibility is defined beforehand and cannot immediately adapt as the market evolves [SMRM20].

Reconfigurable Manufacturing Systems: The concept of RMS was defined by Koren et al. [KHJ$^+$99] as a production system designed to rapidly change its structure, both in software and hardware, based on existing production capacity and functionality. The structure of RMS is dynamic, unlike DMS and FMS and the reconfigurable structure is achieved with modular hardware and software. Resources can be added or removed as needed to increase or decrease production capacity. However, the concept of reconfigurability is often limited to a family of products [ZLGH06], and this primarily hardware-orientated paradigm can lack the intelligence to autonomously adapt to sudden changes [SMRM20, MCS$^+$22].

Autonomous manufacturing systems: Autonomous systems are machines (or groups of machines) that can perform high-level task requirements without being specifically programmed [Bek05]. Such systems automatically accommodate variations in environmental conditions [DCZ$^+$19]. According to Park et al. [PT12], autonomous systems require intelligent and cognitive resources to adapt the system in case of disturbances. This also requires an autonomous control architecture to ensure a proactive decision-making approach.

AMS have the ability to rapidly adapt to changeable environments. Adaptation comprises a wide range of characteristics, from parameter to structural adaptation, and activates all necessary modifications to accomplish manufacturing objectives in highly changeable environments. Adaptation of control logic plays an important role when adapting the behaviour to optimise given goals [PP17]. The system must first sense changes in the environment and actuate accordingly to adopt the best strategy, thereby ensuring that all necessary reconfigurations are carried out seamlessly.

Some of the key technologies of Industry 4.0 and AMS systems are described in the following subsections.

2.1.1 Digital Twins

The nomenclature "digital twin" was coined by Grieves in 2003, who presented it as a conceptual model in the Product Life-cycle Management (PLM) [Gri05]. However, it was not until 2010 that NASA proposed the first definition of the concept as “an integrated multi-physics, multiscale, probabilistic simulation of an as-built vehicle or system that uses the best available physical models, sensor updates, fleet history, etc., to mirror the life of its corresponding flying twin”[SCD$^+$10]. This can be considered the first approximation of a digital replica of a physical twin. Building on this definition, Grieves presented a whitepaper in 2014 which defined the main parts that constitute a digital twin. According to this work, a digital twin is composed of a physical product, a virtual product, and the connection of data and information between the real and virtual spaces.

However, the composition of a digital twin might also include additional components. Tao et al. [TZN19] proposed a five-dimensional digital twin, composed of a physical entity, virtual model, data, services, and connections, as described in Figure 2.2.

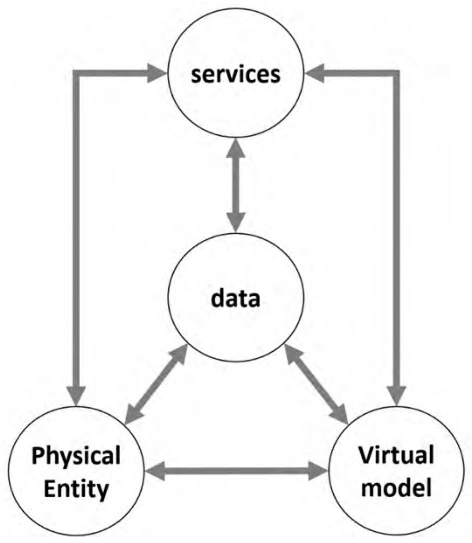


Figure 2.2: Five-dimensional digital twin

- Digital twin data: the digital twin can contain historical and real-time data during the whole life cycle, including, but not limited to real data acquired from the physical entity (e.g. attributes and properties) and its environment (e.g. sensor data), synthetic data generated by the virtual model (e.g. simulation data), and data obtained from the services (e.g. usage data).

- Services: the service interface runs software applications and facilitates interaction and communication between the physical entities and the virtual models, thereby ensuring data accessibility.

- Connections: these are the links that connect the data and information between all the dimensions that comprise the digital twin.

- Physical entity: an entity can be any physical asset, device, system, subsystem, process, person, etc.

- Virtual model: the virtual model might consist of multiple facets to represent the physical entity, comprehensive mechatronic models that include physics-based models, geometrical models, 3D models, logical software models, and control models, among others.

Considering the above definitions, we can say that the digital twin is the virtual representation of a physical asset or system. The definition of the digital twin however is subject to the level of automation of the data flow [KKT+18], which can be classified into "digital model", "digital shadow", and "digital twin", as indicated in Figure 2.3. The Digital Model is a virtual representation of a physical asset, but the communication does not have an active data flow. It is similar to the standard simulation model. The Digital Shadow goes one step further, it has a one-way active communication flow. This means that it can automatically be synchronised with its physical asset, but not the other way around. Lastly, the Digital Twin has a bidirectional data flow. As a result, any change in the physical asset is mirrored in its virtual twin, and vice versa.

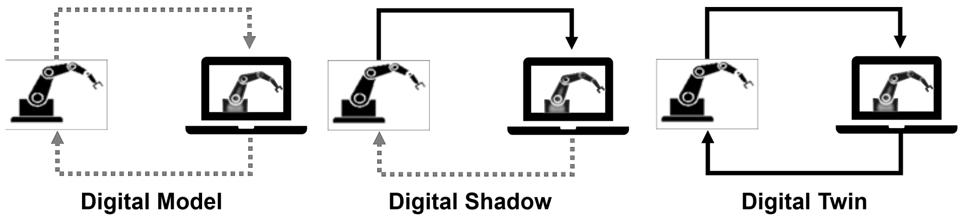


Figure 2.3: Digital twin, Digital Shadow, and Digital Model – adapted from [KKT+18]

The digital twin can be hierarchically categorised, and thus can be composed of lower-level digital twins. Qinglin et al. [QTZZ18] and Tao et al. [TQWN19] divided the digital twin into three different levels: unit level, system level, and System of Systems (SoS) level. The unit level is the smallest unit and represents the field equipment (e.g. sensors, actuators, devices, etc). The system digital twin is composed of multiple unit-level digital twins (e.g. production line, shop floor, factory, etc), and the SoS digital twin is made up of multiple system-level digital twins (e.g. cross-company platform).

2.1.2 Cyber-Physical Systems

The term Cyber-Physical System is attributed to Helen Gill in her 2006 report from a workshop at the National Science Foundation in the United States [Gil06]. They are defined as systems with integrated computational and physical processes [BG11], in which computing devices interact with physical processes via actuators and sensors [Alu15]. Nowadays, CPS are considered fundamental systems for Industry 4.0.

CPS can consist of more than one CPS device or multiple systems of CPS, as illustrated in the CPS conceptual model by Griffor et al. (Figure 2.4 [GGWB17]).

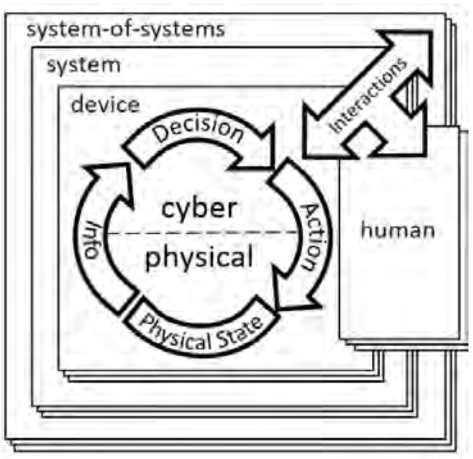


Figure 2.4: CPS conceptual model [GGWB17]

In the manufacturing paradigm, the ISA-95 international standard defines a layered hierarchical automation pyramid [PDK19], shown in Figure 2.5. The information flows from the bottom to the top, and vice versa.

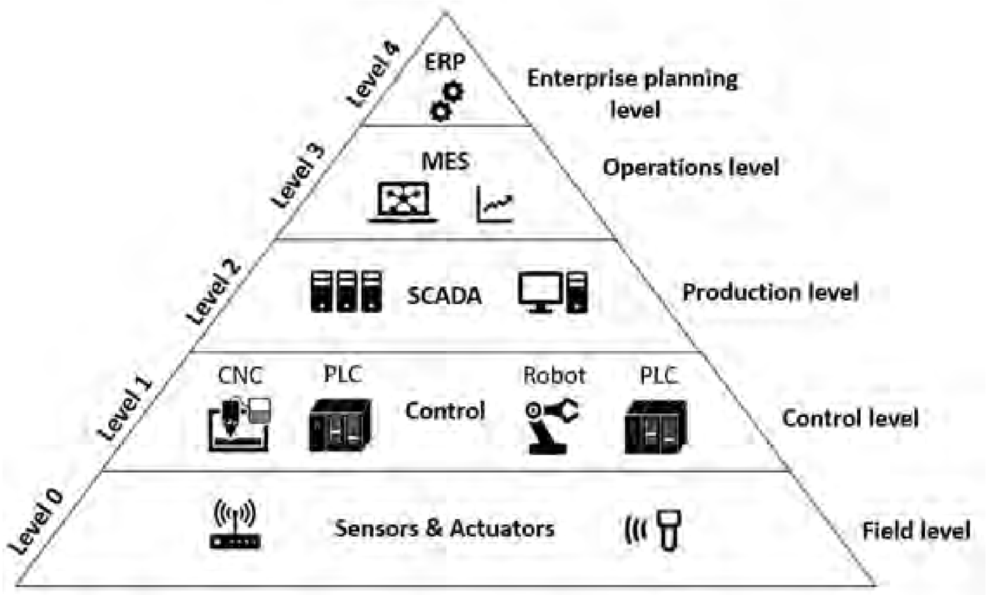


Figure 2.5: Vertical integration based on the ISA-95 pyramid – adapted from [PDK19]

The lowest level is the “field level”, composed of sensors and actuators that interface with physical processes. The “control level” includes control systems such as CNCs and PLCs, which manage and control the resources of the field level. The third level corresponds to the “production level” and mainly refers to Supervisory Control and Data Acquisition (SCADA) Systems that monitor and control the production line. The “operation level”, i.e. Manufacturing Execution System (MES), manages and schedules the production line as determined by the “enterprise planning level”. The highest level of the pyramid refers to business logistic systems, i.e. Enterprise Resource Planning (ERP), which are responsible for order management and business processes.

With the introduction of Industry 4.0 and smart manufacturing systems, however, the ISA-95 hierarchical architecture is evolving towards a distributed control architecture (Figure 2.6), based on modular CPS components [MKB+16], [SGS18].

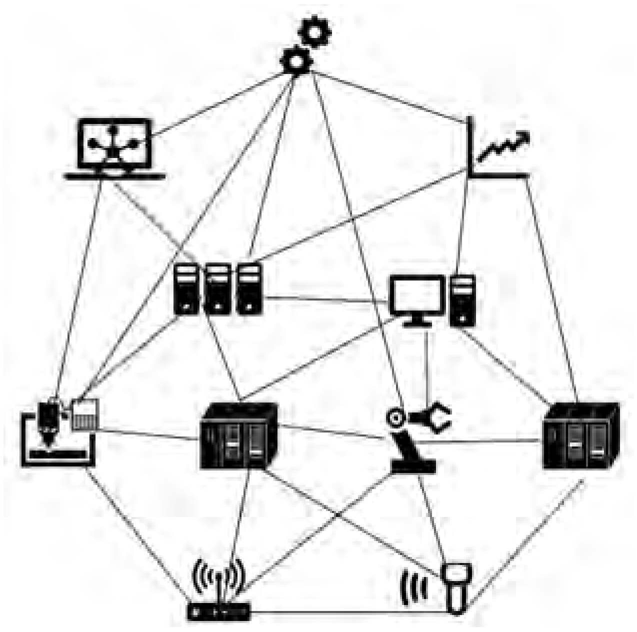


Figure 2.6: Distributed and decentralised CPS architecture

Every modular CPS component is sufficiently autonomous to participate in the value stream of the production process [SGS18]. Thus, CPS components interexchange information with other CPS systems, devices, or services (i.e. MES, ERP), facilitating access to information. This architecture gives rise to a distributed and decentralised control system.

2.1.3 Programmable Logic Controller

PLCs are a type of CPS commonly used to control manufacturing processes in industry [LRP16, AA16, CTC17]. A PLC is an integrated device that consists of a Central Processing Unit (CPU), memory, and input/output ports, which are used to communicate with sensors, actuators and other devices in the system [Mad00]. The control program of a PLC usually consists of a sequence of instructions divided into blocks or sections known as "rungs" of logic. Each rung represents a specific task or set of tasks, that the PLC is programmed to perform.

One of the main characteristics of a PLC is that the software running on these controllers is executed in a cyclical manner [Mad00]. This means that programs are repeatedly executed in loops (called PLC Scan Cycles), performing a series of tasks in a predefined order. Output signals are therefore updated at the end of each scan cycle based on the inputs and previous cycles internal states. This cyclical execution of the control program allows the PLC to constantly monitor the status of the system and make decisions based on the input it receives from sensors and other devices. The PLC can therefore control the operation in real-time and adjust to changes in the system as they occur.

PLCs can operate either as closed-loop or open-loop control systems [Bat96]. In a closed-loop system, there is a feedback path between the controlled system and the PLC. As a result, the PLC continuously updates the input based on the output measured in the system. If no feedback path is required, the PLC is operated as an open-loop control system, sending a predefined set of instructions which are then executed without making any adjustments based on the resulting output.

The software running on these controllers is usually coded using one of the programming languages defined in the IEC 61131-3 standard [TJ10]. These languages can be text-based, like Structured Text (ST) and Instruction List (IL), and graphical-based, such as Ladder Diagram (LD) and FBD.

The majority of PLC suppliers, however, exhibit varying levels of compliance with IEC 61131-3 standard [EMLH10]. To address this issue, the international organisation PLCopen[1] proposed an open XML interface based on the IEC 61131-3 standard. PLCopen XML is an Extensible Markup Language that specifies all the textual and visual notations as indicated in the IEC 61131-3 [WF14]. As a result, PLCOpen XML has been utilised in several studies [EČO$^+$16, SJB18, LJB22a, SFB$^+$15, WF14] to promote interoperability and facilitates the exchange of FBD programs.

FBD is one of the most widely used PLC languages [VHDF$^+$14] in Europe, thus, our work focuses on testing FBD programs, following the PLCOpen XML guideliness.

[1]https://www.plcopen.org/

The following subsection provides more details about the FBD language.

Function Block Diagrams

FBD is favoured for its graphical nature. Programmers can create complex programs by simply connecting a set of predefined blocks together.

In a FBD, Function Calls (FCs) and Function Blocks (FBs) are used to represent specific processes of a control system, and the connections between them indicate the flow of data. The main difference between FCs and FBs lies in the use of internal memory states. While FCs only use input values to calculate outputs, FBs also use internal memory states from previous cycles to generate outputs. In this way, FBs can produce different results with the same input values, depending on the current internal memory states.

The IEC 61131-3 standard defines 10 groups of standard FCs and 5 groups of FBs, as defined in Table 2.1.

Table 2.1: IEC 61131-3 FC and FB groups

IEC 61131-3 FC	**IEC 61131-3 FB**
Data type conversion functions	Bistable elements (flipflop)
Numerical functions	Edge detection
Arithmetic functions	Counters
Bit-shift functions	Timers
Logic functions	Communication
Selection functions	
Comparison functions	
Character string	
Functions for time data types	
Functions for enumerated data types	

2.2 Virtual Commissioning

Traditionally, CPS are tested and validated during the final stage of product development. This can often be error-prone, and can increase development costs and time-to-market. To address these issues, virtual commissioning techniques have been developed to improve system quality and test system behaviour. By using this method, commissioning can be performed earlier in the development process, since it does not depend on any physical resources.

Virtual commissioning is the practice of using virtualisation and simulation technologies that represent the physical system and/or controller in a virtual environment to validate the behaviour of the manufacturing system. Simulation technology allows the development of system engineering alongside concept design. This means that virtual commissioning can be done concurrently or consecutively with virtual design prototyping and engineering, as opposed to traditional commissioning methods. The ultimate goal is to perform partial validations and engineering designs in a virtual environment early in the development process, without the need for a physical version of the system. This significantly improves the design and overall quality of new systems, minimising errors and costs.

There are different alternatives of commissioning, combining real and virtual counterparts of the system under test [LP14], as illustrated in Figure 2.7.

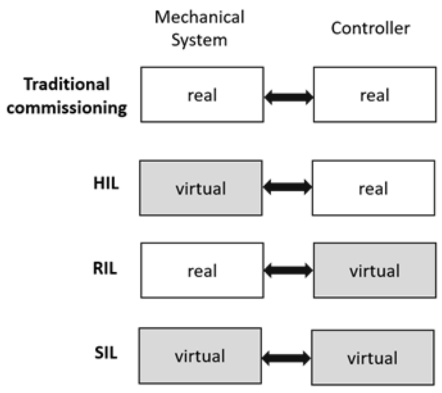


Figure 2.7: CPS commissioning techniques

- Traditional Commissioning: both the mechanical system and the controller are real. The system is not tested until all necessary hardware, including the mechanical system and the controller, is fully installed and connected.

- Hardware-in-the-Loop (HiL): this approach was introduced as an alternative method for reducing delays and addressing issues that may arise before the final commissioning phase [ISS99]. In HiL systems, the mechanical system is replaced with a simulated model that emulates the physical behaviour of the real mechanical system. The real controller (such as a CNC or PLC) is used to control the virtual mechanical system. The goal of this approach is to make the virtual version of the mechanical system available before the real version is completed.

- Reality-in-the-Loop (RiL): in contrast to HiL, which replaces the mechanical system with a simulated model, the RiL approach uses the real mechanical system and substitutes the controller with a simulated control system [AVB99].

- Software-in-the-Loop (SiL): this technique involves using a virtual control system and a virtual mechanical system, and the entire system is virtual. This includes the control system, mechanical system, sensors, actuators, and the process itself. SiL testing allows for the development and testing of the control system and mechanical system in a virtual environment, which can be useful in the early stages [SHC$^+$18].

Commissioning alternatives are on the rise with the introduction of Industry 4.0. This new digital era has brought new technologies such as the digital twin to enhance the virtualisation of factories. The digital twin has emerged as a promising simulation technology, as it mirrors the current state of the physical assets, and can act on them seamlessly. Hence, the use of digital twins can further boost the aforementioned virtual commissioning practices.

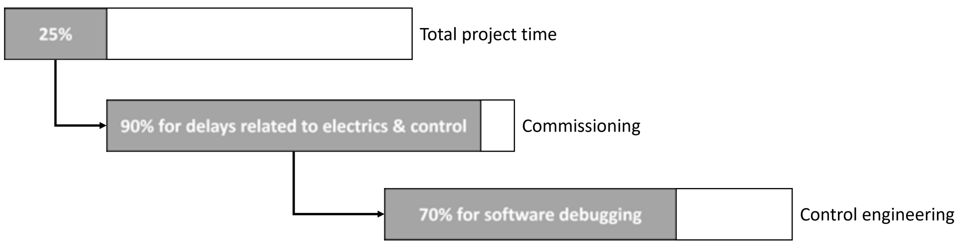

25%
Total project time
90% for delays related to electrics & control
Commissioning
70% for software debugging
Control engineering

Figure 2.8: Contribution of control software to project delay – adapted from [RW07]

The main objective behind these commissioning techniques is to verify that the product meets the established quality standards and satisfies the customer requirements before deploying it into operations. However, bad practices might cause severe delays in this process. As reported in the study of [RW07], Eversheim [EKG90] identified control system malfunctions as a significant cause of delays during commissioning, particularly those related to untested or newly developed control systems. In addition, the commissioning phase of a production system constitutes the 25% of the overall development time, in which 90% of this time is related to delays caused by electric and control devices, and 70% of the delay is attributed to errors in the control software [VB97], as indicated in Figure 2.8.

This indicates that the control software must undergo a comprehensive testing process early in the process to save costs and delays due to software malfunctions. To this end, we present a methodology for testing PLC software using the PLC simulator, which allows us to test the control system, without the need of any physical device.

2.3 Software Testing

Testing is a critical aspect of software development, as it helps to ensure that a system is functioning correctly and meets the specified requirements. There are various testing methodologies that can be employed based on the information available when designing test cases. These methodologies can be classified into three categories: black-box testing, white-box testing, and grey-box testing.

- Black-box testing: black-box testing (also known as functional testing or behavioural testing) is a software testing technique based on software requirements and specifications. This technique lacks of internal knowledge, e.g. internal paths, program structure, or implementation details [DJK12]. The program source code is unknown, hence the name "black-box". Functional testing involves checking the inputs and corresponding outputs to assess that it behaves as expected[NPHS16]. This technique is mainly focused on the functionality of the program, which is based on the output obtained from a given specific input [NT11].

- White-box testing: white-box testing (also known as structural testing) is a software testing technique based on internal paths, logic, and code structure [DJK12]. It uses the program source code as the basis to identify faults within the program code structure, implementation, and logic [Gar17] by testing programming instructions [NPHS16]. White-box testing examines the source code, i.e. execution flow, through coverage (e.g. statement coverage, path coverage, branch coverage) to ensure an acceptable level of testing [RGC14].

- Grey-box testing: grey-box software testing combines the techniques of black-box and white-box testing [CMK+22]. It features basic and limited knowledge of the internal structure of the system [KK+12]. A white-box testing approach is used to design the test cases, whereas a black-box approach is taken to test specified inputs or requirements [DJK12]. Grey-box testing can identify defects that belong to the structure or the application usage [AP12].

The aforedescribed testing methodologies are illustrated in Figure 2.9:

A classification of the most common testing techniques used in black-box, white-box, and grey-box testing is shown in Figure 2.10. The classification is based on the techniques referenced in the following articles: [Gar17, MPZ15, NPHS16, ND12, KK+12, AP12, LM18].

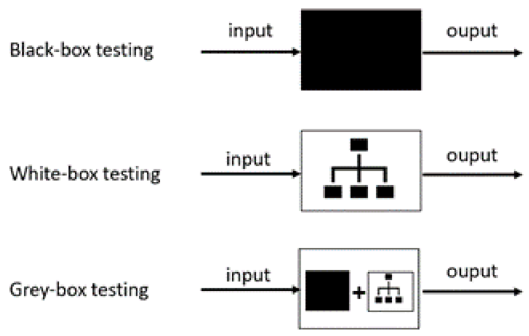


Figure 2.9: Software testing methods

2.3.1 White-box Testing

White-box testing techniques can be used to uncover issues that may not be detectable through functional testing alone. The structural design of the PLC code was known in our case, hence, we applied white-box testing practices, which mainly rely on the structural coverage. In the following subsections, we briefly describe two types of white-box techniques utilised in software testing, which are categorised based on the flow of the program (i.e. control flow and data flow).

Control Flow Coverage Testing

The basic testing model is the flow of control. Test cases are designed to exercise all possible paths through the control structures of the program, and can include the following criteria:

- Statement Coverage (SC): every executable statement in the program is exercised at least once.
- Decision Coverage (DC): every decision in the program is exercised at least once.
- Condition Coverage (CC): every condition in a decision must take all possible outcomes at least once.
- Condition/Decision Coverage (C/DC): combination of DC and CC criteria.
- Modified Condition/Decision Coverage (MC/DC): all condition outcomes that independently affect a decision outcome must be executed at least once, in addition to C/DC.
- Multiple Condition Coverage (MCC): every combination of condition outcomes within a decision is exercised at least once, in addition to MC/DC.

Data Flow Coverage Testing

This method focuses on the data flow of the program under test. It determines data path relationships and dependencies, and the association between the definition of a variable and its uses. It is based on the occurrence of variables, which could be a variable definition or a variable use. Possible metrics include:

- All c-uses coverage: all possible computations affected by a variable definition should be exercised.
- All p-uses coverage: at least one definition clear path from every definition of a variable to every predicate use should be exercised.
- All-uses coverage: all possible uses of a variable should be exercised.
- All-paths coverage: all possible paths should be exercised.
- All d-paths coverage: all possible data paths should be exercised.
- All du-paths coverage: all possible definition-use paths should be exercised.
- All-definition-use-pairs coverage: all possible pairs of definitions and uses of a variable should be exercised.

The described approaches are conventional structural testing criteria based on control flow graphs and procedural languages. However, FBD programs consist of multiple input and output edges, and follow data flow graph models, such as Lustre programs [LP05]. Hence, in Chapter 3 we analysed different coverage testing criteria used in the literature, in which some of them directly employed conventional structural testing criteria by converting FBD programs to control flow graphs, whereas others defined new criteria that suits FBD characteristics.

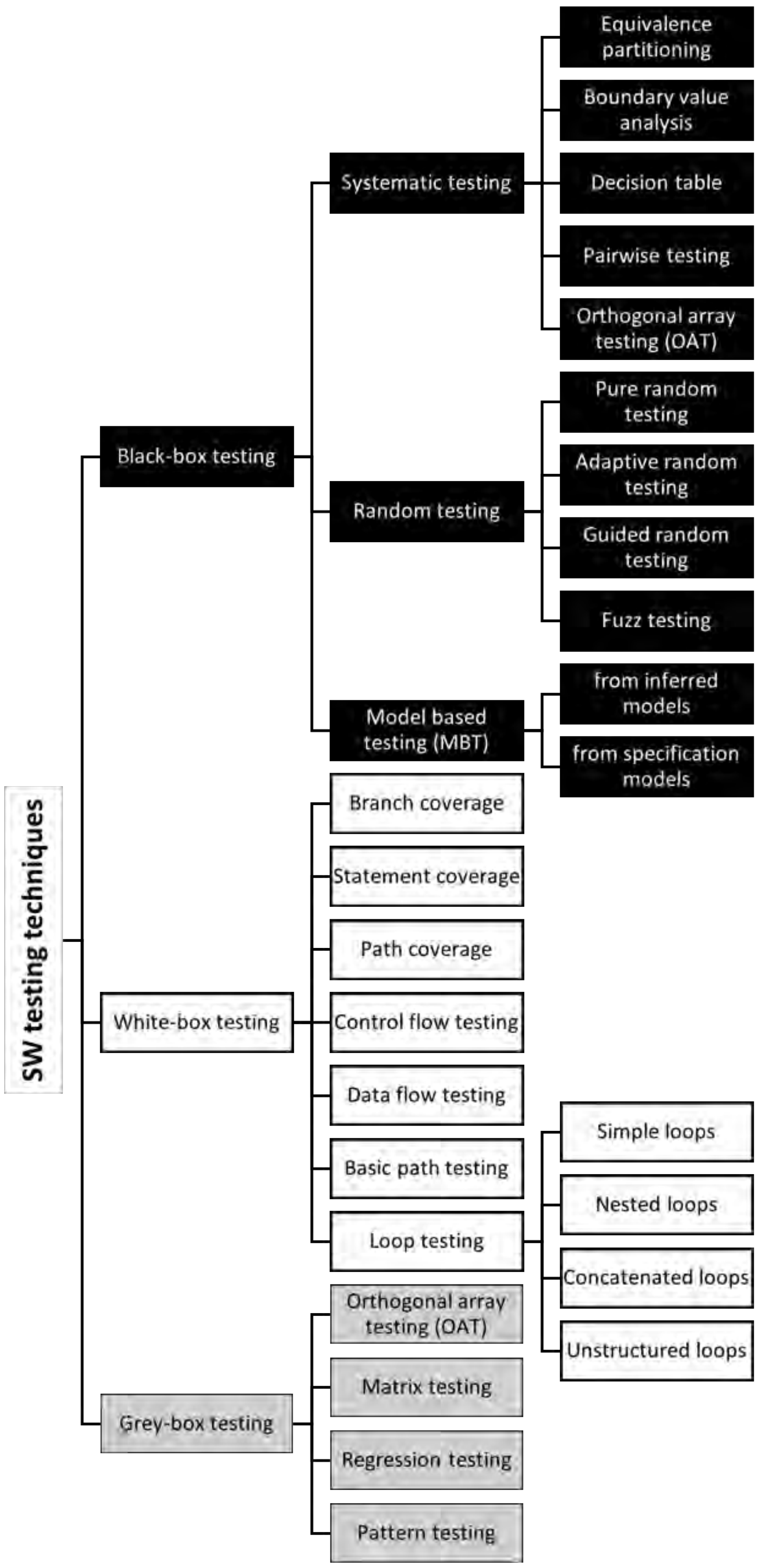


Figure 2.10: CPS testing methods

2.3.2 Mutation-based Testing

Mutation-based testing [PKZ+19] is a software testing technique that can be useful to identify defects or weaknesses as it tests the resilience of the program to faults.

In this technique, the software under test is modified by creating small changes, known as mutations. These mutants are then tested and the results are compared to the expected behaviour of the original code. If a mutant exhibits divergent behaviour, it is said to have been "killed" by the test which indicates that the test has uncovered a problem with the mutant. If the mutant behaviour corresponds to the original program, then the mutant has "survived", and might indicate a weakness in the program.

A wide range of mutant operators is used to reveal different types of faults. These are created typically by replacing a statement or operator with a different statement/operator of the same type. For instance, a logical operator (i.e. OR, AND, XOR) is replaced with a different logical operator, resulting in a modified version of the program. By way of illustration, if the original program contains the statement <*if (a **AND** b)*>, a logical mutant might replace the AND operator with an OR operator, resulting in the modified statement <*if (a **OR** b)*>. Alternatively, the OR operator could be replaced with a XOR operator, resulting in the modified statement <*if (a **XOR** b)*>.

2.3.3 Regression Testing

Regression testing verifies that no faults are introduced in the event of changes to the existing system [LW89]. Regression testing comprises retesting, test selection, test prioritisation, test minimisation, and augmentation approaches [Do16], as indicated in Figure 2.11.

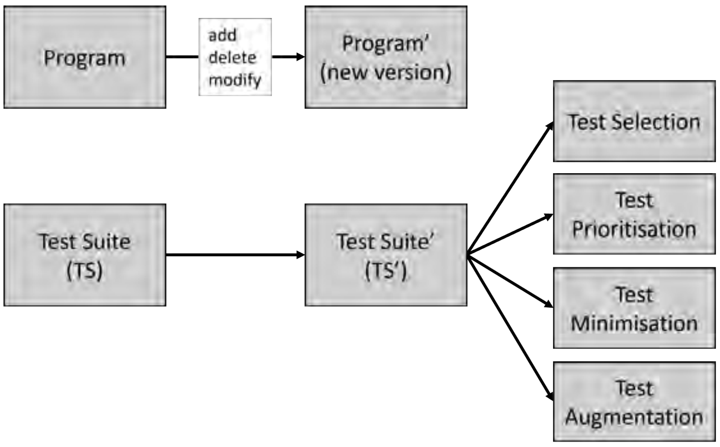


Figure 2.11: Regression testing methods – adapted from [Do16]

Regression test selection and minimisation techniques aim to reduce testing costs by selecting a subset of test cases. Regression test prioritisation, on the other hand, aims to test the whole test suite by ordering the test cases in a cost-effective manner, e.g. detecting faults as early as possible, detecting as many faults as possible, etc.

Test selection techniques can be particularly useful for testing highly configurable CPS, as there are many possible variants and configurations that must be tested [MASE17]. These are cost-effective approaches that reduce the overall time and effort required for analysis, while still ensuring that the most critical aspects of the system are thoroughly tested.

There exists a wide range of test selection techniques for regression testing, as discussed in the literature review carried out by Engströem et al. [ERS10] and Yoo et al. [YH12]. Specifically, search-based approaches have become of great interest within the last years [YH07, RRV19, AWM^{+}19, PODPDL14, HB10]. In the present study, our focus is on search-based test selection, as we aim to guide the search process maximising given objectives.

Search-based Software Testing

Search-based software testing (SBST) can be an effective approach for regression testing, as it can help automatically select and prioritise the test cases that are most likely to uncover defects or problems in the system.

Search-based testing uses search algorithms and heuristics to automatically optimise a testing problem according to a test adequacy criterion, also known as a fitness function. The role of the fitness function is to capture a test objective that makes a contribution to the desired test adequacy criterion. The search algorithm then seeks test inputs that will maximise the predefined test objective, by searching through the space of possible test inputs and configurations.

Genetic algorithms (GAs) are a type of optimisation algorithms that are commonly used in SBST. They mimic the process of evolution and natural selection to find the optimal solution to a given problem. A group of individuals is first randomly generated as the starting population and then evaluated by a selection operator to determine their fitness. The crossover operator combines the traits of these fit solutions (parents) to create new offspring solutions (children). Finally, the offspring solutions undergo a small random change through a process called "mutation". This helps to diversify the population and prevent premature convergence, where the population becomes too homogeneous and stops improving.

Chapter 3

State of the Art

Contents

In this chapter, we review the relevant literature and highlight those studies most related to the topic of this thesis. A critical analysis of the state of the art in the use of the digital twins for virtual commissioning and PLC testing techniques is presented, and research opportunities are identified.

3.1 Digital Twins in Manufacturing

The digital twin has drawn widespread attention with the implementation of Industry 4.0 in the manufacturing sector. Figure 3.1 presents the results of a literature search using the keywords "digital twin" and "manufactur*" in Web of Science (WOS) and Scopus from 2010 to 2022. The results indicate that the concept of the digital twin was not extensively utilised in the manufacturing field until 2016. Since then, the number of publications related to the subject has grown exponentially.

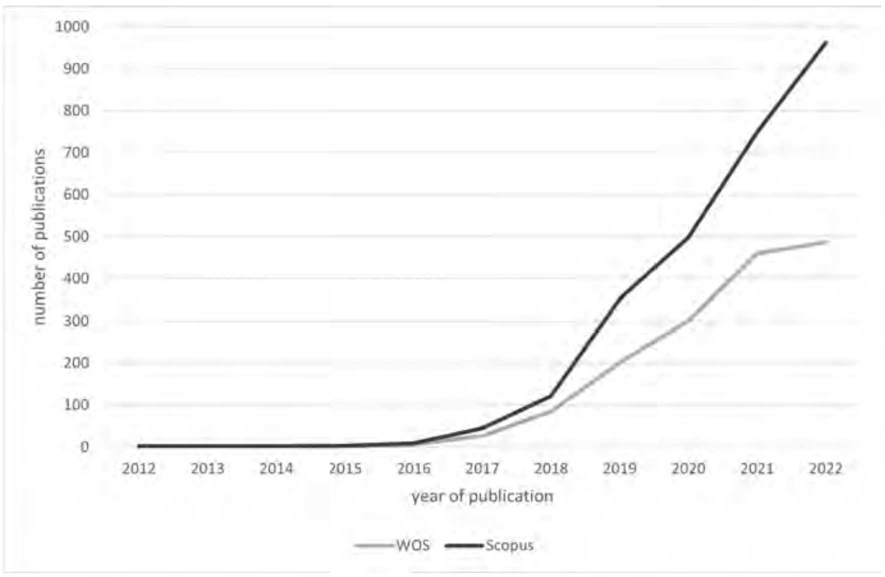


Figure 3.1: Digital twin and manufacturing related publications

Given the considerable interest in this field of study, we aim to determine the primary purpose and usage of digital twins in the manufacturing industry. To this end, we evaluate digital twin-based applications throughout the entire lifecycle of an asset, starting from design and development, and finishing with the operational and maintenance phases. Our goal is to gain a deeper understanding of how digital twin technology is being used, as well as identify gaps and areas for improvement in the current usage and application. To this end, we conducted a snowballing search in the study of Jones et al. [JSN+20], and the resulting studies are described in Table 3.1.

Table 3.1: Publications related to digital twin applications

Ref.

Use case	**Application**	**Field**	
[BMKW18]	Yes	Production planning based on the automated creation of the digital twin of a body-in-white production system	Manufacturing
[SBL$^+$20]	Yes	Assembly-commissioning for High Precision Products	Manufacturing
[WWY$^+$20]	Yes	Predictive CPS remanufacturing	Manufacturing
[LZY$^+$19]	Yes	Holistic online parallel controlling	Complex systems
[DEA$^+$19]	Conceptual	Online optimisation of the system operation based on context data	Manufacturing
[KCK19]	Yes	Runtime controllability verification of a control command	Manufacturing
[ZQZ$^+$20]	Yes	Optimal state control framework	Manufacturing
[ZS20]	Yes	Controllability of the physical layer	Manufacturing
[LLY$^+$20]	Yes	Reconfiguration of the manufacturing system for reacting to the changeover of the product order	Manufacturing
[ZYC19]	Yes	Real time status warning of the production process	Manufacturing
[BS19]	Yes	Real time monitoring of the process	Manufacturing
[MPPU19]	Yes	Additive manufacturing supply chain through the exploitation of blockchain	Additive, Manufacturing
[TLNN18]	Yes	Learning environment for engineering education	Manufacturing
[LLKM18]	Conceptual	A versatile learning environment to facilitate collaboration between industry and academia	Manufacturing
[YTYT17]	Yes	Simulation model for runtime experiments	CPS, Manufacturing
[GBK$^+$16]	Conceptual	Online planning to predict system behaviour	CPS, General
[SAMW17]	Conceptual	Geometrical variations management	Manufacturing

[SKH18]	Conceptual	Designing production and logistics systems	Manufacturing
[EE18]	Yes	Automated generation of digital twins in a secure environment	Manufacturing
[USL+17]	Conceptual	Learning factory	Manufacturing
[SWCL17]	Conceptual	Real-time geometry assurance	Manufacturing
[TIES11]	Conceptual	Structural life prediction	Aerospace
[LML+17]	Yes	Health monitoring	Aerospace
[Rod17]	Conceptual	Simulation modelling paradigm	Manufacturing
[BGS+18]	Yes	Security evaluation of specific functionalities	Manufacturing
[BHO+18]	Yes	Productivity analyser for production process improvement	Manufacturing
[GZSZ19]	Yes	Modular and flexible factory design	Manufacturing
[AGS17]	Yes	Cloud-based smart reconfiguration platform	Manufacturing
[CSCL17]	Yes	Real-time monitoring and tool wear prediction for prognosis and optimisation	Manufacturing
[BCMV17]	Yes,	Decision-making autonomy and self-adaptation of machines to solve material handling problems	Manufacturing
[MAH18]	Conceptual	Model-based definition to incorporate behaviours and product characteristics into CAD model.	Model-based Engineering
[TWE18]	Conceptual	Traceability and improvement of the future wiring harness design based on digital twin data	Automotive
[TSL+19]	Conceptual	Redesign the next generation of bicycles, simulations, design optimisation	Manufacturing
[DB18]	Conceptual	Privacy enhancement mechanism	Automotive
[DE+18]	Conceptual	Root cause analysis and product quality monitoring	Manufacturing
[ZLCX18]	Yes	Design approach for Smart Product Service System innovation	Smart PSS
[AGN18]	Yes	Reconditioning of a machine	Manufacturing, Automotive

[WW19]	Yes	Recycling, recovery and remanufacturing services for electrical and electronic equipment waste	
[LZLC19]	Yes	Rapid individualised design of Automated Flow Shop Manufacturing System	Manufacturing
[ZZT18]	Conceptual	Equipment Energy Consumption	Manufacturing
		Management improvement (monitoring, multilevel and multistage analysis, behaviour analysis, statistical analysis, parameter optimisation, scheduling optimisation, machine tool upgrading)	
[DR18a]	Yes	Modelling and simulation-based engineering approach based on digital twins	Complex Systems
[DR18b]	Yes	Simulation-based verification with experimentable digital twins	Complex Systems
[KMT+17]	Yes	Factory Telemetry for virtual reality simulations	Manufacturing
[MRNF18]	Conceptual	Asset-related decision-making (maintenance, system life cycle mirroring, optimisation)	Manufacturing
[LWIL20]	Conceptual	Digital twin-driven virtual verification	Manufacturing
[DDG+18]	Yes	Simulation and design of a production line	Manufacturing
[HG+17]	Conceptual	Traceability of product data management over the whole life cycle by the implementation of the blockchain	Automobile
[SPAR18]	Yes	EDTs as a new structuring element for simulation-based systems engineering	Manufacturing
[TLHN20]	Yes	Digital twin-based virtual commissioning for CNC machine tools	Manufacturing
[TW20]	Yes	Reconfiguration process of a warehouse with the digital twin	Manufacturing
[Jan18]	Yes	Virtual commissioning of heavy machine tool based on the digital twin	Manufacturing
[EA20]	Yes	Virtual commissioning of an industrial wood cutter machine	Manufacturing

The selected applications were classified according to the most suitable stages of the lifecycle during both the development and operational phases of a CPS. The development phase involves the design, engineering, and integration phases, and requires testing practices for the verification and validation of the model before being released. Once the development process is completed, the product is realised into operations and service. The latter however could be fed back to the development phase, by following a DevOps approach [QES20], ultimately enabling a continuous and agile production system. To this end, we classified the use of the digital twin according to the most suitable stages, as indicated in Table 3.2. This helps us have a better picture of existing practices and identify potential needs.

Table 3.2: Classification of digital twin applications

	Development			**Operations**	
Citation	**Design**	**Engineering**	**Commissioning**	**Operations**	**Service**
[BMKW18]	X				
[SBL+20]			X		
[WWY+20]				X	
[LZY+19]				X	
[DEA+19]				X	
[KCK19]				X	X
[ZQZ+20]				X	
[ZS20]				X	
[LLY+20]				X	
[ZYC19]					X
[BS19]					X
[MPPU19]					X
[TLNN18]					X
[LLKM18]					X
[YTYT17]				X	
[GBK+16]				X	
[SAMW17]	X				
[SKH18]	X				
[EE18]	X	X	X		
[USL+17]					X
[SWCL17]	X				
[TIES11]				X	
[LML+17]					X
[Rod17]		X			
[BGS+18]	X	X	X		
[BHO+18]				X	
[GZSZ19]	X				
[AGS17]	X	X		X	

[CSCL17]				X	X
[BCMV17]				X	
[MAH18]		X			
[TWE18]	X			X	
[TSL+19]	X	X			
[DB18]					X
[DE+18]					X
[ZLCX18]					X
[AGN18]		X	X		
[WW19]					X
[LZLC19]	X				
[ZZT18]				X	X
[DR18a], [DR18b]	X	X	X		
[KMT+17]					X
[MRNF18]				X	
[LWIL20]	X	X	X		
[DDG+18]	X	X			
[HG+17]	X	X			X
[SPAR18]	X	X	X		
[TLHN20]		X	X		
[TW20]		X	X		
[Jan18]		X	X		
[EA20]	X	X	X		

The results of this categorisation are depicted in Figure 3.2 and Figure 3.3, which provide an overview of the digital twin throughout different stages of the lifecycle.

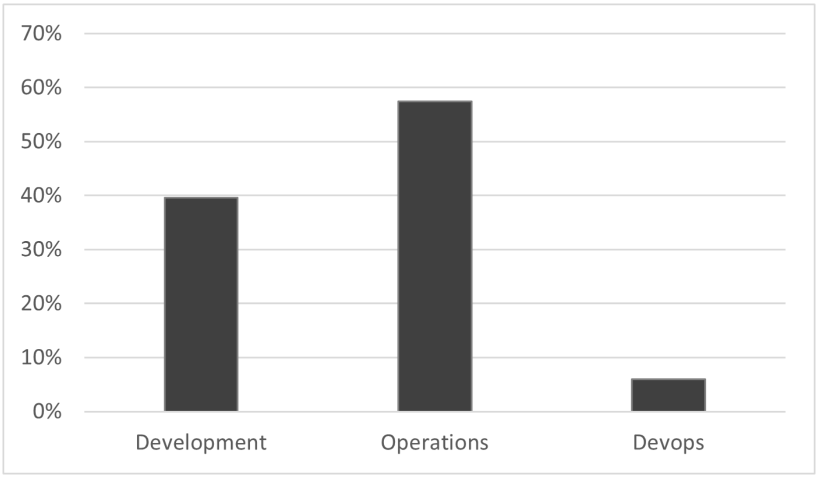


Figure 3.2: Classification of digital twin applications throughout the lifecycle

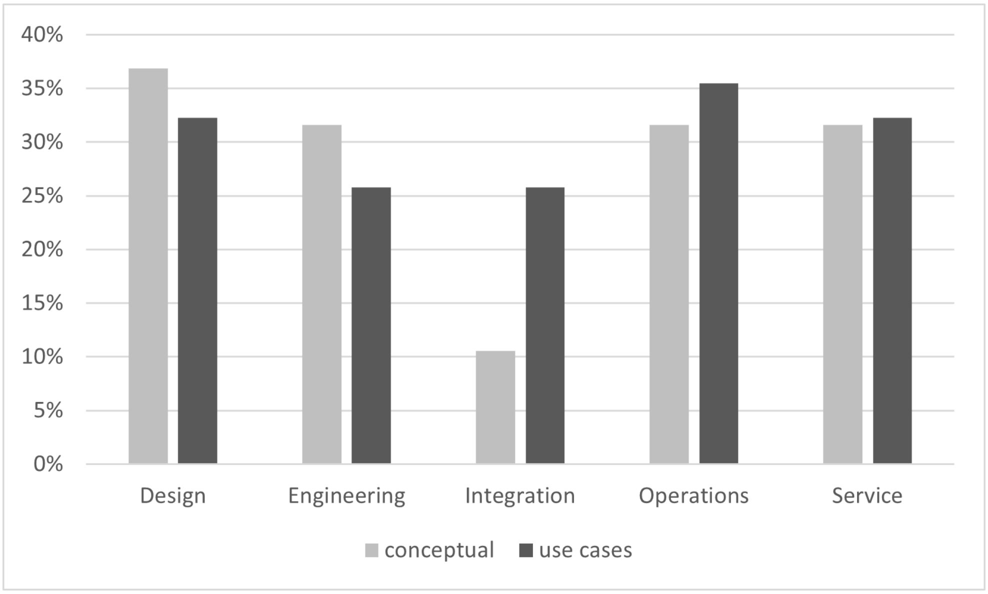


Figure 3.3: Classification of digital twin applications into development and operations

On the one hand, Figure 3.3 reveals that the digital twin is mostly employed during the operations, service, and design phases, but its application during the integration phase is limited. This suggests untapped potential for the digital twin in this area, and highlights a pertinent field of study. On the other hand, Figure 3.2 shows that most applications and use cases are in the field of operations and development, with operations as the primary area of focus. However, limited research has been carried out into the potential use of digital twins in both the development and operations cycles (i.e. DevOps).

DevOps approaches are being used widely in software development, specifically for web-based applications, delivering faster applications and automating the development and deployment of web-based applications from end to end [EGHS16]. In recent years, the CPS domain has also started to leverage DevOps practices. Garcia and Cabot applied DevOps practices at the model level for the very first time [GC19], and a few studies explored DevOps for model-driven engineering of CPS [WBCW20, CW20], and continuous deployment, monitoring and validation of CPSs [GAA+21]. More recently, Hasselbring et al. [HHL+19] proposed industrial DevOps as an approach to introduce methods and culture of DevOps into industrial production environments, and Hegedũs et al. [HVF] presented a new workflow for the development and deployment of industrial IoT and CPS devices based on DevOps practices. Similarly, Hugues et al. [HHHY20] presented TwinOps as a process that unifies Model-Based engineering, Digital twin and DevOps practices for the engineering of CPS. Given the interest in DevOps practices in the CPS domain and industry, the manufacturing industry could benefit from implementing a DevOps approach to create a continuous production system. This indicates an area of opportunity for future research, in which the digital twin could be used to adopt DevOps practices in the manufacturing sector.

3.2 Virtual Commissioning

Commissioning, the last step in the development process, is crucial to ensure the system is properly verified and validated before it is deployed into operations. One effective way to achieve this is through the use of virtual solutions which simulate the behaviour of the system in a virtual environment. Virtual commissioning has been a subject of study for the past decades [HSMP10], and has attracted great interest with the emergence of Industry 4.0.

Research has shown that virtual commissioning can significantly reduce commissioning time by up to 75%, as shown in [KPL+11]. Nevertheless, with the increasing digitisation of industry, new technologies have become available that can be used to further enhance the digitisation of commissioning practices.

3.2.1 Digital Twin-based Virtual Commissioning

Digital twin technology synchronises the virtual model of a system or process with its physical counterparts in real-time. This helps address the issue of model inconsistencies that can arise in traditional virtual commissioning practices, potentially saving time and effort (as described in [TLHN20]). As such, Xueming et al. [SBL+20] introduced a digital twin-based approach for assembly commissioning to improve efficiency and quality. The implementation of this technology resulted in a 37.5% reduction in assembly time compared to traditional methods, and a significant improvement in assembly quality. Additionally, Schamp et al. [SHC+18] conducted a small-scale test of digital twin-based virtual commissioning and found that it led to a 75% reduction in debugging time, and a 31% improvement in quality.

Digital twins have been also exploited to test and validate configuration changes before they are deployed, which can optimise the continuous integration process. For example, in a study by Talkhestani and Weyrich [TW20], the use of a digital twin for testing the reconfiguration of an intelligent warehouse resulted in a reduction in reconfiguration process time of up to 58%. Other studies have also demonstrated the potential of digital twins and virtual commissioning for simulating and optimising the reconfiguration of production systems, such as a modular reconfigurable plug-and-play conveying system (Hofmann et al. [HUL$^+$18]) and a newly configured system of a fluid power machine (Alt et al. [AMMS18]). All of these studies highlight the potential benefits of using digital twins and virtual commissioning to improve the efficiency of reconfiguring production systems.

In the machine tool industry, the potential application of digital twins to HiL and SiL machine testing and system validation early in the process has been demonstrated. Shen et al. [TLHN20] proposed a digital twin-based virtual commissioning approach for CNC machine tools. This methodology involves creating a unified model using various engineering modules and a mapping strategy. Several virtual commissioning practices have also been developed that use a variety of models, including automation, electric, and kinematic models. For example, Janda [Jan18] designed a HiL solution that utilised a Siemens NX 3D mechatronic model, a Simit communication bridge, and a Sinumerik 840D control system to create a digital twin of a heavy machine tool. In contrast, Edgar and Montero [EA20] developed a SiL solution using Siemens NX MCD, Simit, PLCSim advanced, and a TIA Portal automation solution. Table 3.3 summarises some of the commercially available simulation and emulation tools which support virtual commissioning.

The recently launched Siemens SINUMERIK ONE is the only holistic solution that integrates all simulation models (mechanical, automation, control, etc.) into a single environment or hardware. However, this solution is vendor specific as it is limited to Siemens products. This presents a significant drawback, as there is no option to integrate components from different manufacturers.

3.3 Testing of FBD-based PLC Programs

Traditionally, functional testing and validation of PLCs has been a time-consuming and labour-intensive process, as it often requires manual testing using the physical hardware [VHFST15, SCN$^+$21]. However, there has been a growing interest in automating PLC testing to improve efficiency and reduce the amount of manual work involved. Research has focused on developing methods and tools to automate the testing process.

There have been many efforts to use formal methods to verify the correctness of FBD programs. A common approach is to use model checking, which is a technique which formally verifies the behaviour of a system against a formal specification. In this regard, some researchers [SF11, dSdABG$^+$08, EDB$^+$13] have used timed automata for real-time verification by converting FBD to UPPAAL [BDL04] timed automata – an integrated tool environment for modeling, validation, and verification of real-time

Table 3.3: Simulation and emulation tools for virtual commissioning

Tool	**Characteristics**
Siemens NX MCD	- High level of integration between different engineering disciplines
Siemens SINUMERIK ONE	- Capacity to create the digital twin from one engineering system
Siemens Process Simulate	- Validation of assembly lines - Optimisation of production lines
Siemens Plant Simulation	- Analysis of material flow - Discrete event simulation - Optimisation of material handling, logistics operations, and resource utilisation
Dassault Delmia	- Precise virtual production system - Simulation and optimisation of manufacturing assets with production planning
ISG Virtuos	- Real-time control (<1ms) - Operations with controllers and field buses
Simumatik 3D	- Open emulation platform - Server-client architecture - Compatible with different PLCs - Ideal for education but also suitable for professionals
Emulate3D	- Compatible with different PLCs - Offline digital twin - Support to virtual and augmented reality
XCelgo Experior	- Compatible with different PLCs - .NET-based development environment - Physics simulation, discrete event simulation, 3D graphics

systems. Others have focused on using binary decision diagram-based symbolic model verifiers (SMVs) to verify the correctness of PLC programs [PMLK13, PE10, JJC$^+$10, YCJ08]. Additionally, some works have used Petri nets as a tool for the verification of PLC programs [BMMP00, VDJB14]. However, it is still challenging for engineers without expertise in formal methods to write specifications for PLC programs.

In light of this, several research efforts have attempted to automate the generation of IEC 61131-3 code using standardised modelling languages such as Unified Modelling Language diagrams (UML). UMLs are a widely-used graphical representation for specifying, visualising, and documenting the design of software systems. For instance, Vogel-Heuser et al. [VHWK05] described a method for automatically generating IEC 61131-3 code from UML diagrams. Similarly, Hussain et al. [HF06] and Hametner et al. [HKVH$^+$13] proposed a dynamic test case generation approach that is based on UML state charts and the IEC 61499 standard [Vya09]. Mapping UML diagrams to FBD can be complex, however, because it involves translating the graphical representation of the system in the UML diagram into the specific instructions and logic of the IEC 61131-3. This requires a detailed understanding of both the UML diagram and the IEC 61131-3 FBD language, as well as the relationships between different elements in the diagram and the corresponding blocks. Moreover, it can be challenging to map all the FBD information in a clear and accurate way.

On the other hand, the automation and testing of PLC programs is an active area of study. Functional testing, which involves conducting tests based on the requirements of a control system, is a common approach for testing PLC programs. Traditionally, functional testing of PLCs has been carried out manually, which is time-consuming and error-prone. This is particularly challenging in the case of highly reconfigurable PLCs, as exhaustive testing is necessary each time there is a new modification in the code. Automation in PLC testing practices is therefore crucial.

Fernandez et al. [FBM13, BVBS19] have been actively working on the automation and testing of PLC programs. In particular, they analysed algorithmic formal verification and automatic testing for PLC validation and testing. Their latest study [BVBS19] was primarily focused on continuous integration, which is considered to be a key component in ensuring an agile software development and testing process.

In this context, Talkhestani et al. [TJL$^+$19, TBSW20] proposed the use of a digital twin as an agile technology to automatically generate control codes for new machines, using the Anchor Point Method. The method focuses on the automatic detection of changes and the adjustment of the digital twin model. The resulting PLCs codes are generated using Siemens TIA Portal. Similarly, Koziorek et al. [KGK$^+$19] suggested using TIA Portal as a tool for automating PLC code generation and testing activities.

The aforementioned studies have made substantial progress towards automating PLC testing to reduce manual labour. However, to the best of our knowledge, they have not put the spotlight on the internal design of PLC programs, which may not be efficient at identifying implementation errors. As a result, automated structural testing of PLC programs has emerged as an area of interest in the field of industrial automation. By automating structural testing, the efficiency and effectiveness of PLC testing can be increased by reducing the time and effort needed for testing, and by offering a more comprehensive evaluation of the control system.

3.3.1 Control Flow Testing

Some authors have focused on control flow-based test data generation, similar to those implemented in procedural languages.

In this regard, Enoiu et al. [ESP13, EČO+16] presented a structural test generation approach for FBD programs using the UPPAAL model checker. They introduced a tool called CompleteTest, which can automatically convert FBD programs into timed automata models and generate test suites. They also defined testing criteria for FBD programs based on control flow-based coverage criteria, such as decision coverage, condition coverage, and MC/DC. In a similar vein, Wu and Fan [WF14] designed an automatic test case generator for FBD programs using the UPPAAL model checker, named FPCCTestGen. Using this approach, they automatically transformed FBD programs into timed automata models to generate test cases. However, both these approaches require the conversion of FBD programs to intermediate state chart models or control graphs. This adds some complexity and does not accurately represent data flow relationships of the respective FBD blocks [HKVH+11].

Similarly, Lahtinen [Lah14] developed a model-based test generation approach, in which FBD programs are manually modelled into binary-decision diagrams. Test requirements are then automatically generated and transformed into temporal logic formulas for property checking. This approach also offers automatic test generation for FBD programs, but first they need to be manually modelled into binary decision diagrams, which can be an error-prone and time-consuming task.

Jee et al. [JYC05] applied control flow coverage criteria to evaluate FBDs by transforming them into control flow graphs. They generated test cases that meet the All-edges coverage criteria. Interestingly, the results indicated that control flow graphs were not able to effectively represent the data flow characteristics of FBDs.

Other studies, such as Simon et al. [SFB+15] and Bohlender et al. [BSF+16] used the Arcade.PLC model checker to generate test cases according to line or branch coverage. However, it is important to note that once again, these methods require the user to establish formal models before test cases can be generated.

In an alternative approach, Ulewicz et al. [UVH18] identified untested behaviour without the need for formalised requirements or simulations. The study was carried out based on statement coverage. However, this methodology entails the addition of code instrumentation, which can be a cumbersome and laborious task to implement in practice.

Thus it can be seen that various researchers have proposed different approaches for test generation of FBD programs. However, all of these present certain limitations such as the complexity in converting FBD to intermediate state chart models or control graphs, manual modelling, and instrumentation of the code. Further research is therefore needed to address these limitations and develop more efficient and practical approaches for the test generation of FBD programs.

3.3.2 Data Flow Testing

The field of data flow-based coverage testing for FBDs has been widely researched in recent years, with a focus on developing more accurate coverage criteria that can better represent the characteristics of FBD programs.

One of the most significant contributions to this research area was made by Jee et al. [JYCB09]. They established new coverage criteria for FBDs based on formal definitions of the internal design characteristics of FBDs. These new criteria are founded on the data paths and Data Path Conditions (DPC) of the FBD to check the data flow between the different blocks in the program, in contrast to control flow-based coverage metrics. These criteria include Basic Coverage (BC), Input Condition Coverage (ICC), and Complex Condition Coverage (CCC), as described below:

- BC coverage: all distinct paths of the program must be exercised at least once. BC coverage requirements therefore are defined by the DPC of all distinct paths in the program under test.
- ICC coverage: all distinct paths and all possible variations (i.e. true, false) for each Boolean input edge must be tested.
- CCC coverage: all distinct paths and all possible variations for each Boolean edge must be tested.

In contrast, Maruchi et al. [MSS14] and Lee et al. [LKY21] presented a new MC/DC-like Structural Coverage Criteria for FBD based on data path definitions. They also presented a new criterion, named Propagation Toggle Coverage (PTC) to check that the changes of all edges propagate to outputs independently. However, these criteria could not assess all possible data paths of the FBD. Jee et al. criteria results to offer a more comprehensive assessment of the FBD. Building on this work, Jee et al. [JYCB09] and Song et al. [SJB16] further advanced the field by defining calculation libraries for representative IEC 61131-3 based FCs and FBs. With this foundation, Jee et al. [JSC$^+$14] developed an automated test generation tool called FBDtester, which conforms to FBD-specific coverage criteria without the need for any intermediate models. However, this tool was limited to one PLC scan cycle, which made it unable to properly represent the characteristics of FBD programs that require various scan cycles or iterations. As a result, most of the FBD blocks such as timers, counters, etc., in which outputs are dependent on internal status, could not be tested. To address this limitation, Jee et al. and Song et al. [SJB18] presented the next version of the FBD tester, named FBDTester 2.0. This tool can automatically generate coverage-based test cases for FBD programs using various scan cycles. The given program and test requirements were defined from coverage constraints, and test suites were generated to satisfy the test requirements using the Yices SMT (Satisfiability Modulo Theories) solver [DDM06]. However, the solver used in this tool could not solve non-linear equations, such as divisions, multipliers arithmetic equations, etc.

Coverage-based test generation has been proposed as a way to internally check the structure of FBD programs in order to detect potential faults. However, coverage calculation can be time-consuming as a large set of requirements must be defined and assessed first to see if the defined assertions are satisfiable. This can be a challenge for large and complex FBD programs. Additionally, the limitation of not being able to solve non-linear equations in the solver used, also limits the applicability of the test generation tool. These gaps could be addressed in future research to improve the efficiency and effectiveness of coverage-based test generation for FBD programs, or to explore other testing methods that could be used in conjunction with coverage-based testing to enhance the detection of faults in FBD programs.

3.3.3 Mutation-based Testing

Ensuring the correctness and reliability of FBD programs is a crucial task. Mutation testing is a widely used technique for fault detection in software engineering [JH10, PKZ+19]. However, there is little research when exploring the use of mutation testing to identify defects in FBD programs.

Enoiu et al. [ESv+16] proposed for the very first a mutation-based test generation technique for FBD testing. They introduced six FBD specific mutation operators to generate mutants. Their work relied on the UPPAAL model checker to automatically generate mutation-based tests based on a combined timed automata model that contained all the mutants and the given program. As a way to evaluate the effectiveness of the testing methods, they used faults that were manually inserted by industrial engineers. The results showed that mutation-based test generation can achieve better fault detection than coverage-based test generation, but not as good as manually-created test suites. However, mutation-based test suites are more efficient as they require less testing time compared to manual test case generation.

Recently, Liu et al. [LJB22b] proposed the MuFBDtester, which is inspired on their previously developed tool FBDTester 2.0. It utilises the same environment and SMT solver-based technique as FBDTester 2.0. The tool automatically generates

SMT constraints by merging the original program and its mutants, and it employs 13 different mutation operators. The proposed mutation-based testing was able to detect a vast majority of artificial faults, including those that were not identified by test suites generated to meet coverage criteria. The experimental results showed that the MuFBDTester performed more effectively and efficiently on subject programs than the approach proposed by Enoiu et al. [ESv$^+$16]. Additionally, the MuFBDTester supports more standard blocks and mutant operators than Enoiu et al.'s work.

As future lines, Enoiu et al. [ESv$^+$16] proposed using advanced mutation operators such as the feedback loop insertion operator, which is specifically designed for PLC programs. They also suggested exploring higher-order mutations.

Mutation-based test generation techniques showed promising results in terms of effectiveness. However, there are still gaps in the current research that need to be addressed in future studies, such as reducing time and cost associated to these techniques. Additionally, to the best of our knowledge, there is still a significant gap in integrating PLC testing techniques in industrial environments.

3.4 Search-based Test Selection

In recent years, search-based methods have gained traction in the area of test case selection to cost-effectively optimise tests. In particular, test selection has received significant attention within the field of regression testing [ERS10], in which retesting all test cases is time-consuming and costly.

Yoo and Harman [YH07] were the first to present a multi-objective Pareto efficient search-based algorithm. They employed both two-objective and three-objective formulations. In the former, they used code coverage as a measure of test adequacy, and execution time as a cost. The latter combined coverage, cost, and fault history. Since then, multiple studies have focused on multi-objective search-based algorithms for CPS testing.

Among others, Pradhan et al. [PWAY16] described a search-based approach for multi-objective test selection by defining one cost measure (time difference) and three effective measures (mean priority, mean probability, and mean consequence). Similarly, Lachmann et al. [LFN$^+$17] proposed a multi-objective test case selection technique, in which they defined seven different objectives for black-box testing. Arrieta et al. [AWA$^+$18] also presented a black-box-based multi-objective test selection approach to cost-effectively optimise tests. Specifically, they defined five effectiveness measures (i.e. instability, discontinuity, growth to infinity, input-based test similarity, output-based test similarity) and one cost measure (test execution time). In total, 15 different objectives were derived by combining up to two effective metrics with the cost. They used the popular Non-dominated Sorting Genetic Algorithm II (NSGA-II) search-based algorithm, the results of which outperformed the baseline Random Search (RS) performance. All of these studies focused on combining different fitness metrics to cost-effectively select test cases.

More recently, [PODPDL14, AVAS23] proposed seeding strategies to introduce some diversity when initialising search populations. On the one hand, Panichella et al. [PODPDL14] proposed a diversity-based genetic algorithm that seeded the initial population with orthogonal arrays. To this aim, they proposed a diversity-based genetic algorithm (DIV-GA), in which the main loop of the NSGA-II is combined with the diversity-preserving mechanism. They found that the presented diversity algorithm outperformed the NSGA-II. On the other hand, seeding strategies to cost-effectively select test cases were introduced by Arrieta et al. [AVAS23]. They defined four seeding strategies to initialise the population of search algorithms. The approach was integrated into the PlatEMO [TCZJ17] open-source platform, which is a well-known multi-objective search library for MATLAB. They assessed four search algorithms (i.e. NSGA-II, IBEA, SPEA2 and PESA-II) with 6 fitness functions. The results showed that seeding strategies have great potential in multi-objective test selection problems.

Given frequently changing market needs, regression testing has been applied to highly configurable CPS [ASEZ17, PSS$^+$16] and product lines. Wang et al. [WAG15] applied search algorithms to minimise the test suite for testing a product, while preserving fault detection capability and testing coverage of the original test suite. They defined four effectiveness measures (i.e. test minimisation percentage, pairwise coverage, fault detection capability, and average execution frequency), and one cost measure (i.e overall execution time).

An iterative test allocation for testing CPS product lines was defined by Markiegi et al. [MASE17]. Their methodology is based on product selection and prioritisation regression techniques, followed by an iterative test allocation mechanism that assigns a set of test cases to each product. The iterative testing mechanism relies on multi-objective search algorithms with the aim of optimising the fault detection capability, test execution time, and test case appearance frequency when testing product lines with multiple configurations. In a similar vein, A. Arrieta et al. [AWSE19] also applied a search-based approach for testing CPS product lines in a cost-effective manner. They empirically analysed various search-based algorithms to reduce fault detection time, simulation time, and requirements covering time at each test level. The results demonstrated that the selected search algorithms performed better than the RS algorithm.

Search-based multi-objective test selection strategies have been demonstrated to cost-effectively optimise CPS testing. However, no studies have been found that investigate optimisation methods for selecting test cases for PLCs which also consider cost-effectiveness.

3.5 Critical Analysis of the State of the Art

In this section, we present a critical analysis of the current state-of-the-art in testing highly configurable PLCs in the context of Industry 4.0, with the aim of identifying potential research opportunities from the sections reviewed in the literature.

Digital Twins in manufacturing: The emergence of Industry 4.0 has led to the development of new technologies and has created further opportunities for research. One such area is the use of virtual technologies for commissioning, which has been shown to significantly reduce time and costs. Several studies have proposed digital twin-based virtual commissioning as a promising approach for the future of commissioning systems. This idea has been discussed in [TLHN20, SBL+20, SHC+18].

Although the digital twin has generated great interest in the manufacturing sector, it has yet to be widely adopted in practice. The most striking result to emerge from the review of the literature (Table 3.2), is that most applications fall into either development or operations, and very few attempt to address both cycles (*Gap 1: limited use of the digital twin for DevOps*). This opens up new research opportunities, in which the digital twin could be used to leverage DevOps practices in manufacturing, as suggested by Hugues et al. [HHHY20], Hegedũs et al. [HVF], and Hasselbring et al. [HHL+19]. The second major finding is that there are relatively few studies focusing on the integration phase, covering CPS commissioning and testing practices, among others (*Gap 2: lack of digital twin-based applications in the integration phase*).

Virtual commissioning: With regard to virtual commissioning tools, we identified 9 different products: Siemens NX MCD, Siemens Numerik One, Siemens Process simulate, Plant simulation, Dassault Delmia, ISG Virtuos, Simumatik 3D, Emuate3D, and XCelgo Experior. Despite the wide range of solutions commercially available, Sinumerik is the only holistic solution integrating various simulation models in the same environment. However, it is not vendor agnostic and is limited to the Siemens products family (*Gap 3: no vendor-agnostic and holistic solution for virtual commissioning*).

Some authors have gone beyond virtual commissioning, by using the digital twin to test and re-commission new configurations of CPS [TW20, AMMS18, HUL+18].

Effectively, CPS are modular systems that can have multiple configurations by combining different modules in the context of plug and produce. Scant attention has been paid to the main actors in industrial automation, however, specifically PLCs in this context (*Gap 4: limited use of the digital twin for commissioning and reconfiguring PLCs*).

Testing of FBD-based PLC Programs: Testing PLCs is essential as these are the core control system in the manufacturing industry. Given the existing disruptive market needs, PLCs are likely to undergo further and frequent changes in the code. To ensure the robustness and reliability of PLCs, thorough testing is necessary to identify and address any potential weaknesses. For this reason, there is an increasing interest in automating the integration and commissioning of PLCs to reduce manual labour [BVBS19, TJL+19, TBSW20, KGK+19]. In terms of PLC software testing methods, formal approaches, coverage-based testing, and mutation-based testing techniques are among the most commonly employed.

Numerous studies have attempted to utilise formal methods for verifying the correctness of FBD programs. One popular approach is to use model checking, in which several authors have employed timed automata models [FBM13, SF11, dSdABG+08, EDB+13], SMV [PMLK13, PE10, JJC+10, YCJ08], Petri nets [BMMP00, VDJB14], and UML diagrams [HF06]. However, these remain challenging for engineers without expertise in formal methods, as they necessitate a comprehensive understanding of IEC 61131-3 standard and the selected modelling language. Furthermore, a model-to-model transformation process is required, resulting in a loss of information (*Gap 5: complex formal verification methods for FBD programs*).

With regard to coverage-based testing techniques, two types of coverage-based testing methods were identified: control flow-based and data flow-based. On the one hand, [ESP13, EČO+16, WF14, HKVH+11, Lah14, JYC05, SFB+15, BSF+16, UVH18] presented control flow-based test data generation techniques for FBDs, similar to those implemented in procedural languages. Nevertheless, there are some limitations to this approach, such as the complexity of converting FBDs to intermediate state chart models or control graphs, manual modelling, and instrumentation of the code (*Gap 6: complexity of control flow-based testing methods for FBDs*). On the other hand, [JYCB09, LKY21, JYCB09, JSC+14, SJB18, DDM06] focused on data flow-based methods, with the goal of providing more accurate coverage criteria that can better reflect the characteristics of FBD programs with no further model-to-model transformations (addressing *Gaps 5-6* challenges). In particular, the FBDTester 2.0 presented by [SJB18] yielded promising results. The implemented solver could not address non-linear equations, however, which limits the applicability of the tool (*Gap 7: limitation of the FBDTester 2.0 in handling non-linear arithmetic functions*). Moreover, coverage calculation can be time-consuming as a large set of requirements must first be defined and evaluated to determine if the assertions are satisfactory.

Mutation testing is another testing technique, widely used in software engineering for fault detection [JH10, PKZ+19]. However, few studies have explored the use of mutation testing to identify defects in FBD programs. [ESv+16] and [LJB22b] have presented promising results, but there remain gaps in the current research that need to be addressed, such as reducing the time and cost associated with these techniques (*Gap 8: lack of research on cost-effective mutation testing techniques*).

Additionally, to the best of our knowledge, none of the aforementioned testing studies have been implemented yet in industrial environments (*Gap 9: lack of PLC testing techniques in industrial environments*).

Search-based test selection: In this context, search-based multi-objective test selection strategies have been shown to be cost-effective in optimising testing for CPS. However, to the best of our knowledge, there are no optimisation methods for selecting test cases for the PLC application domain (*Gap 10: lack of search-based test optimisation methods for PLCs*). This could be due to the complexity and unique characteristics of PLC systems.

Hence, this critical review of the state-of-the-art has identified a clear need for efficient and effective test case selection methods to test FBD programs, so as to reduce time and costs. Furthermore, no studies have been identified in which coverage-based and mutation-based testing techniques have been implemented in industrial PLCs or industrial PLC simulation environments. To address these gaps, the present body of research focuses on:

- Research Focus #1: Designing a vendor-agnostic virtual commissioning framework that enables the continuous integration of PLCs following DevOps principles (*Gaps 1-4*).
- Research Focus #2: Optimising the existing data-flow-based coverage and mutation testing techniques to address a vast majority of IEC 61131-3 function groups (*Gap 7*).
- Research Focus #3: Optimising tests to cost-effectively detect implementation errors on FBDs (*Gap 8, Gap 10*).
- Research Focus #4: Presenting a methodology to automate testing techniques on industrial PLCs (*Gap 9*).

Chapter 4

Industrial Survey

Contents

In this chapter, we present the conducted industrial survey as a complementary study to the state-of-the-art. The survey aimed to gain further insights into the challenges faced by the industry in adopting digital twin and virtual commissioning solutions, placing particular emphasis on *Research Focus #1* and *Research Focus #4* raised in Section 3.5.

According to the literature, the digital twin is widely recognised as the next wave in modelling and simulation [RVWLB15]. Digital twins are well positioned to enhance virtual commissioning practices, as they can address synchronisation issues and misalignments between virtual representations and the physical system [QES20]. While virtual commissioning typically consists of various simulation technologies commissioned in silos by distinct engineering disciplines, the digital twin integrates all these models into a single framework.

Although a wide variety of tools and technologies are commercially available, no existing solution integrates multiple domain simulation and emulation technologies into a single and unified platform. As a result, companies are reluctant to invest in simulation technologies for virtual commissioning, because the efforts required will not yield a sufficient ROI. This poses a serious obstacle to the adoption of virtual solution strategies.

Hence, in this chapter we present the results of a survey carried out to assess industrial and academic readiness, and understand existing needs and challenges in adopting virtual commissioning and the digital twin. This first-hand information is essential to better align *Research Focus #1* and *Research Focus #4* with the industry needs.

4.1 Survey Design

This section describes the design and structure of the empirical survey we conducted at an industrial and academic level. Section 4.1.1 defines three sets of Research Questions (RQs) related to virtual commissioning, digital twins, and testing procedures. Section 4.1.2 sets out the target population. Lastly, the structure and question flow is detailed in Section 4.1.3.

4.1.1 Research Questions

The RQs were categorised into three groups. The first group were designed to determine industrial readiness for the use of virtual commissioning, and identify the main gaps. In total six RQs were developed to gain insight into industry experience with conventional commissioning practices, and understanding of the concept of virtual commissioning, including the benefits and challenges. These research questions are directly linked to *Research Focus #1*.

- RQ1.1. What are the main challenges in traditional commissioning?
- RQ1.2. What is understood by "virtual commissioning" in industry?
- RQ1.3. Does virtual commissioning involve different engineering disciplines?
- RQ1.4. Is virtual commissioning carried out at earlier stages of the development process?
- RQ1.5. What are the perceived benefits of performing virtual commissioning?
- RQ1.6. What are the main challenges posed by virtual commissioning?

The second set of RQs focused on the evaluation of industrial and academic readiness in terms of the digital twin. To this end, four RQs were formulated, including the definition of the digital twin, the degree of implementation level, and identification of the main benefits and challenges. These research questions are directly linked to *Research Focus #1*.

- RQ2.1. What is understood by a digital twin?
- RQ2.2. What is the degree of implementation of the digital twin in industry?
- RQ2.3. What are the main challenges posed by the digital twin?
- RQ2.4. What are the perceived benefits of the digital twin?

The third group consisted of four questions, focusing on testing procedures, the need for automated testing practices, and the primary challenges associated with testing. These research questions are directly linked to *Research Focus #4*.

- RQ3.1. What are the tests carried out during the commissioning process?
- RQ3.2. Are these tests automated?
- RQ3.3. Is there a need for automating tests?
- RQ3.4. What are the main testing challenges in the development and operation of a machine tool?

4.1.2 Target Population

The target population of the survey comprised both academic researchers and industrial practitioners with experience in virtual commissioning projects. The population included some of our industrial partners and stakeholders, ESRs, and beneficiaries from the DiManD project.

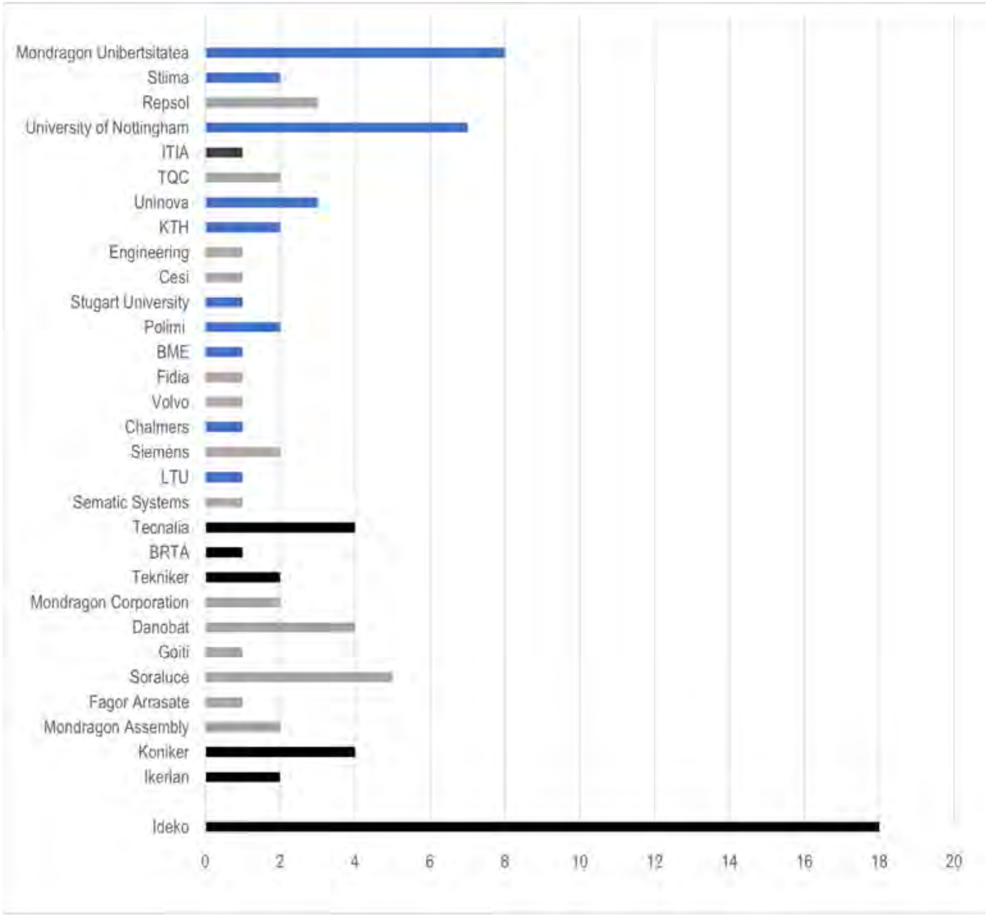


Figure 4.1: Survey target population

This survey was conducted as part of the secondment activity at IDEKO. Hence, we first surveyed IDEKO employees to gather feedback on the quality of the questions. This step helped further refine the RQs, which were then shared with members of Mondragon Corporation, a federation of worker cooperatives located in the Basque Country, Northern Spain. Once the results were confirmed to satisfactorily address the RQs, the survey was further circulated to a diverse range of companies and industrial partners of IDEKO. These included global leaders in the machine tool sector, such as Siemens, Fagor, and Volvo. In addition, the survey was also completed by the beneficiaries and ESRs of the DiManD network.

In total, 31 organisations were involved in the survey, from which 87 individuals were contacted. These organisations were categorised as academia (in blue), research centre (in black), or industry (in grey), as illustrated in Figure 4.1. Nevertheless, the research centres under study were R&D companies and thereby, they were combined with the industry category to simplify the analysis process and provide a clearer distinction from academia.

4.1.3 Survey Structure

The survey consisted of 39 questions categorised into six groups, as depicted in Figure 4.2. These included questions related to individual backgrounds, organisational information, and technical questions addressing the three sets of research questions (i.e. traditional and virtual commissioning, digital twins, and testing procedures during commissioning). A copy of the questionnaire can be found in Appendix A.

- Background questions (Q1-Q6): this block includes questions related to individuals, such as gender, years of experience in the manufacturing industry, years of experience in the organisation, and expertise in different technological areas.
- Organisational questions (Q7-Q8): these encompass questions about the sector and type of organisation of the surveyed participants, i.e. company, academia, or research centre.
- Technical questions: this section comprises 31 questions on topics such as traditional commissioning, virtual commissioning practices, utilisation of the digital twin, and testing procedures:
  - Traditional commissioning (Q9-Q12): the purpose of this block is to address the difficulties commonly faced in conventional commissioning in response to RQ1.1.
  - Virtual commissioning (Q13-Q23): these questions address RQ1.2, RQ1.3, RQ1.4, RQ1.5, and RQ1.6. Topics covered include the concept of virtual commissioning, its benefits and challenges, the purpose of its application, the engineering disciplines involved in the process, and the level of complexity associated with virtual commissioning.
  - Digital twin (Q24-Q32): this set of questions answer RQ2.1, RQ2.2, RQ2.3, and RQ2.4, and are related to the definition of the digital twin, its advantages and challenges, and the primary purpose of implementing a digital twin. However,

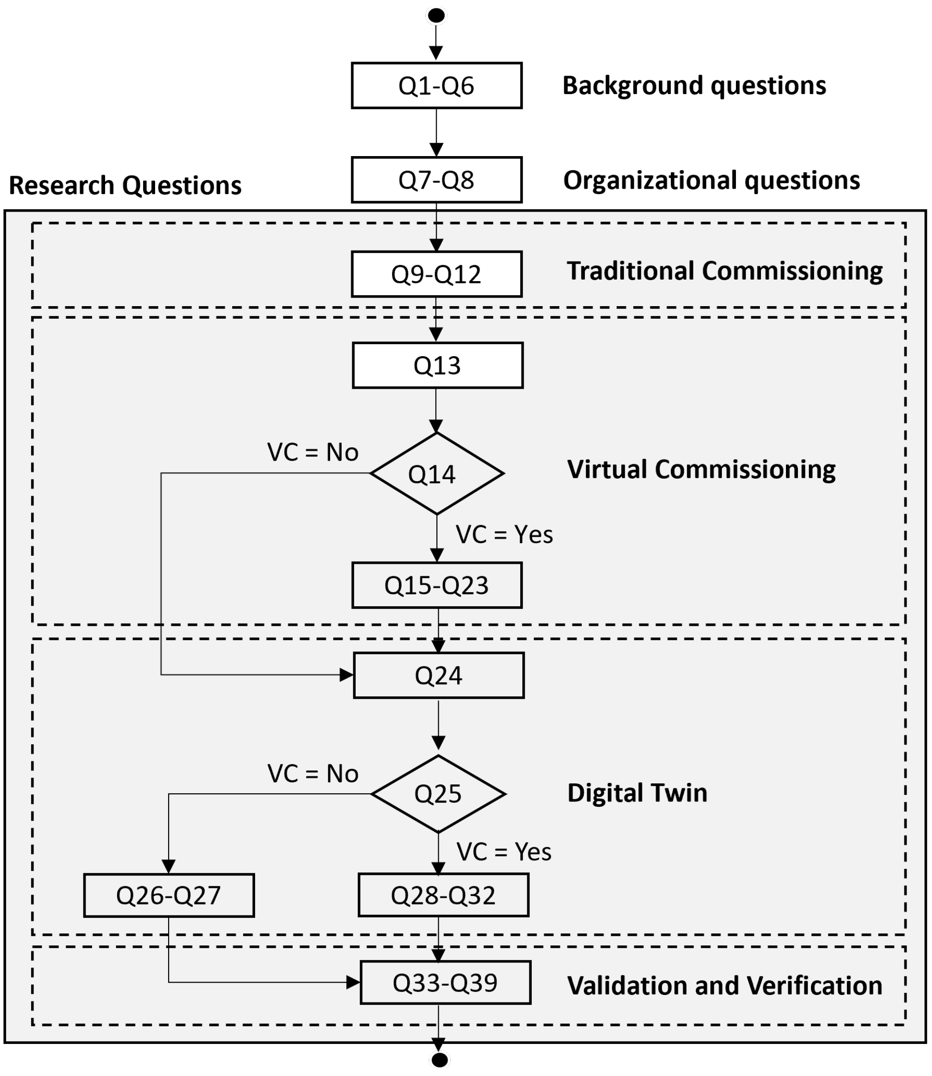


Figure 4.2: Survey structure

some of the questions were conditional and thus could only be answered by participants who had implemented this technology.

- Verification and validation tests (Q33-Q39): this block addresses RQ3.1, RQ3.2, RQ3.3, and RQ3.4. These questions are related to the testing methodology, the engineering disciplines involved, and any challenges faced when testing CPS.

4.2 Results

We obtained a total of 26 responses to the survey, of whom 12% were female and 88% male. Specifically, 23% of the participants had a background in mechanical engineering, 23% in electronics, 19% in automation, 12% in computing science, and 8% in ICT. The remaining 15% were from mechatronics, industrial organisation, telecommunications, and systems engineering. Eighty-one % of the participants were experienced professionals with over 10 years of experience. Half reported experience in the machine tool manufacturing sector.

The survey was carried out as per the ethics requirements specified in the DiManD project (Appendix B). To this end, the survey data was treated as confidential and anonymous, and the results do not disclose any individual or sensitive organisational information.

4.2.1 RQ1.1. What are the main challenges in traditional commissioning?

The objective of RQ1.1 is to address the challenges associated with traditional commissioning, which is typically carried out at the end of the development process. This approach is believed to be error-prone as the system is not tested until everything is fully installed. Thus to gain insight from industry and academic experience related to traditional commissioning we asked the participants to identify the most significant commissioning issues from a number of options reported in the literature [LFM$^+$19]: time to market, limited option for error correction, unexpected problems arising from errors in previous development stages, difficulties in testing the electrical system, and development costs.

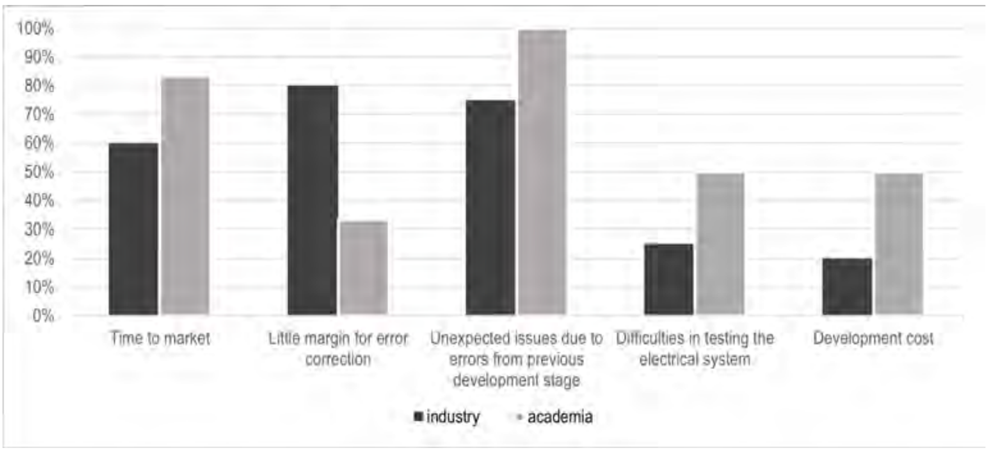


Figure 4.3: Challenges in traditional commissioning

Figure 4.3 highlights the principal difficulties in traditional commissioning for both industry and academia. Unforeseen complications resulting from errors in previous development stages is identified as the most critical. Another major concern for

80% of researchers and 60% of industry practitioners is time to market. Moreover, 80% of industrial respondents believe there is little room for error correction. This result reinforces *Ch4 (time to market and costs)*, and identifies a clear need for virtual commissioning to accelerate time to market, minimise commissioning errors, and ultimately, reduce overall costs resulting from time delays and unexpected issues

4.2.2 RQ1.2. What is understood by "virtual commissioning" in industry?

The practice of testing system behaviour with a virtual machine model before connecting it to the real system is termed virtual commissioning. However, depending on the virtualisation solution, virtual commissioning could take place in a number of testing levels: HiL (real controller and virtual plant), RiL (virtual controller, real plant), and SiL (virtual controller and virtual plant). It is therefore crucial that everyone employs the same terminology and shares the same conceptual definition. RQ1.2 examines this issue by asking survey respondents to select the definition closest to their particular virtualisation solution.

RQ1.2 also seeks to broaden the definition of virtual commissioning, by highlighting its key advantage: performing a series of collaborative verification tasks across various engineering disciplines and throughout the entire development process.

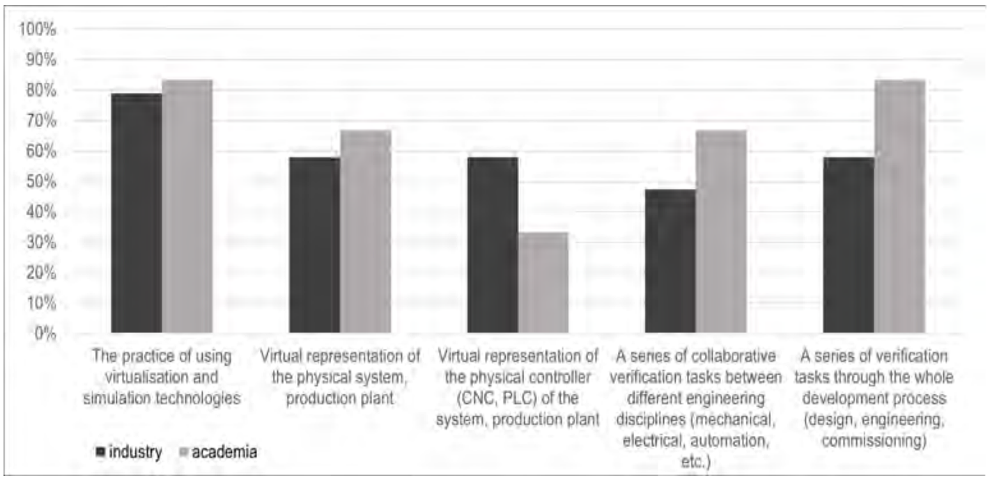


Figure 4.4: Virtual commissioning definition

Figure 4.4 depicts the industry and academic definition of virtual commissioning. The percentage of the population is set out on the *Y* axis, and the surveyed statements of virtual commissioning are on the *X* axis.

The figure shows that 80% of the industry primarily identifies virtual commissioning as the use of virtualisation and simulation technologies. Of these, 58% consider the virtual representation of the production plant and/or controller. Those surveyed in academia, however, also highlighted the performance of a series of collaborative verification tasks across engineering disciplines and throughout the entire development process. In contrast, only 47% and 58% of industry respondents selected these aspects. This shows that industrial practitioners still lack sufficient understanding of the full advantages of virtual commissioning, in which partially supports *Ch6 (difficulty in quantifying the ROI).*

4.2.3 RQ1.3. Does virtual commissioning involve different engineering disciplines?

RQ1.3 addresses the statement "virtual commissioning facilitates a series of collaborative verification tasks between multiple engineering disciplines". To this end, industrial and academic respondents were asked to identify the disciplines involved in virtual commissioning. Figure 4.5 depicts the engineering disciplines on the *X* axis (mechanical, electronic, CNC control and automata, software, robotics, design, telecommunications, and systems engineers), and the percentage of the industry and academic population on the *Y* axis.

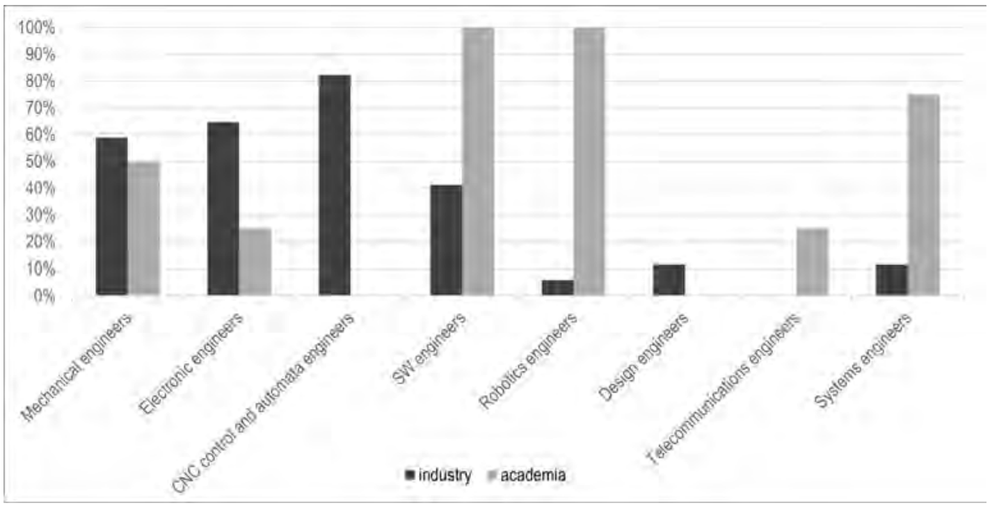


Figure 4.5: Virtual commissioning practitioners

The results show a marked disparity between industry and academia perceptions. Industry respondents selected CNC control and automation engineers (82%), electronic (65%) and mechanical engineers (59%), as disciplines most involved in virtual commissioning. In contrast, those surveyed from academia believe that software and robotics engineers are critical (100%), while 75% also believe that systems engineers are required. However, it is clear that a collaborative environment is crucial, as indicated in *Ch5 (lack of collaborative and holistic solutions)*, combining the diverse skill sets required for successful virtual commissioning.

4.2.4 RQ1.4. Is virtual commissioning carried out at earlier stages of the development process?

The statement "virtual commissioning facilitates a series of virtual verification tasks throughout the whole development process" is covered in RQ1.4. Figure 4.6 illustrates industry and academic perceptions of the stages of the development process (*X* axis) in which virtual commissioning is mostly conducted. The figure shows that industrial

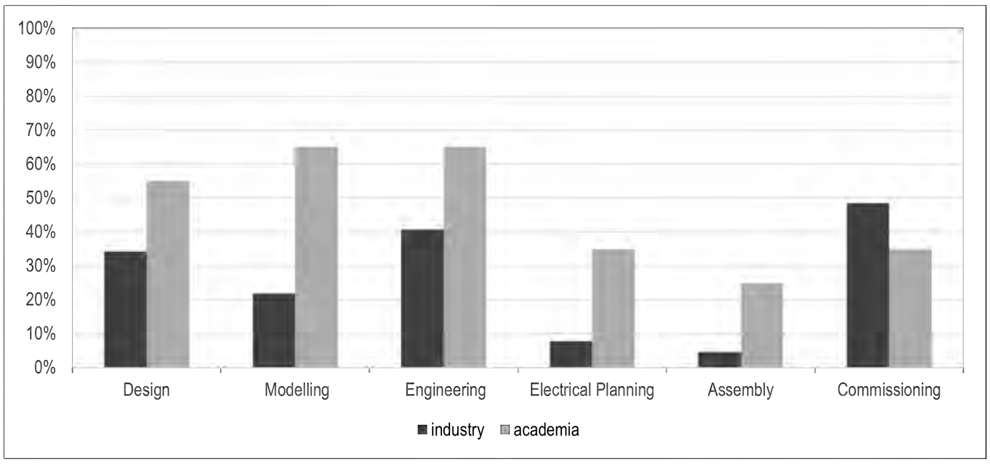


Figure 4.6: Virtual commissioning through the development process

practitioners show a marked preference for virtual commissioning at the final stage of the development process, which leads to *Ch4 (time to market and costs).* Academic respondents, in contrast, concur with the idea that virtual commissioning is carried out at earlier stages including design, modelling, and engineering. These results reveal a significant gap between academic and industry perceptions of virtual commissioning during the development process.

4.2.5 RQ1.5 and RQ1.6. What are the main benefits and challenges posed by virtual commissioning?

RQ1.5 and RQ1.6 concern the benefits and challenges of virtual commissioning, and the responses to these questions are summarised in Table 4.1. These are partially aligned to *Ch6 (difficulty in quantifying the ROI)*.

Table 4.1: Benefits and challenges of virtual commissioning

Benefits	**Challenges**
Testing without disrupting the physical system	Standardisation and interoperability
Higher test coverage	Software functionality limitations
Ability to reconfigure virtually, and test and validate the system before physical commissioning	System variability
Reduced time to market, errors, and costs	Validity and fidelity of simulation models
Better root cause analysis in case of failures	ROI: efforts vs benefits
Higher confidence during FAT and SAT	Traditional mindsets
Quality improvement in final product	Scope limited to PLC testing
	Lack of required simulation fidelity details

4.2.6 RQ2.1. What is understood by a digital twin?

The concept of the digital twin has evolved over time and there is no single definition of the term. RQ2.1, therefore, asks respondents to identify and define the key aspects and components of a digital twin. Figure 4.7 plots the definition of a digital twin as selected by respondents.

Both academia and the industry consider the digital twin as the virtual representation of a physical asset, and 67% of researchers believe it also represents a process, human, performance, etc. Interestingly, only 50% of both sectors selected the practice of using virtualisation as simulation technologies.

4.2.7 RQ2.2. What is the degree of implementation of the digital twin in industry?

Industrial readiness for implementation of the digital twin is measured in RQ2.2. Participants were asked to mark the degree of implementation from 0 (not implemented yet) to 10 (fully implemented). The results are presented in Figure 4.8, which categorises the degree of implementation as low (1-3), medium (4-6), or high (7-10).

As can be seen, 55% of industry practitioners have not yet implemented or have a low implementation level of the digital twin. Partial implementation sits at 30%, and only 15% confirm a high degree of implementation. These results would suggest

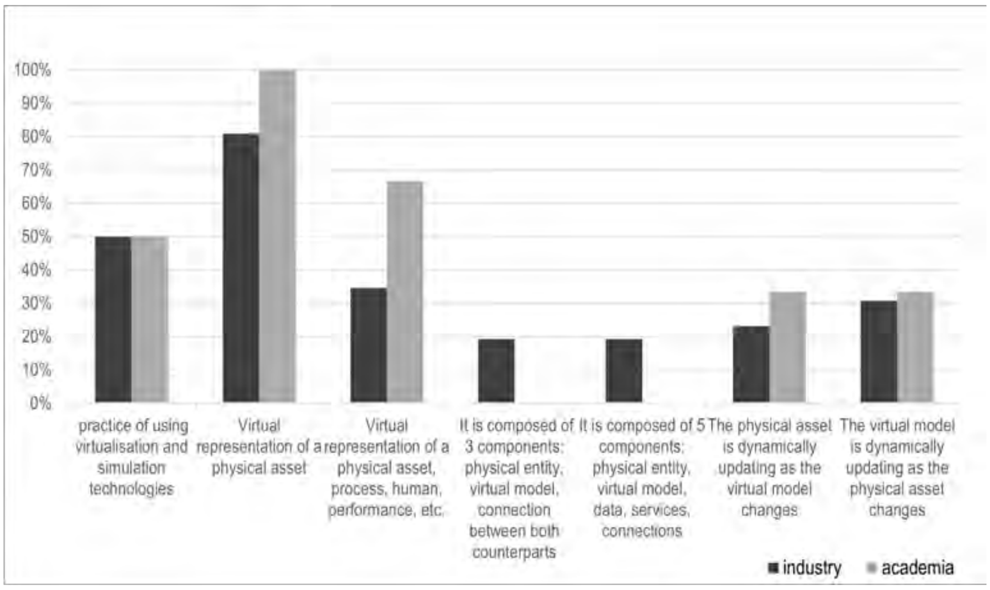


Figure 4.7: Digital twin definition

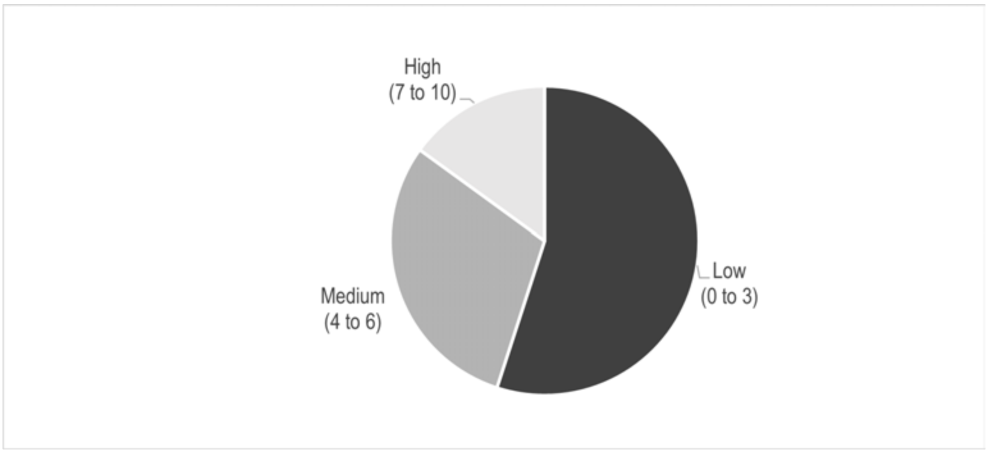


Figure 4.8: Degree of implementation of the digital twin in industry

that the implementation of the digital twin for commissioning purposes remains a work in progress, as indicated in *Gap 2 (lack of digital twin-based applications in the integration phase*) identified in Section 3.5.

4.2.8 RQ2.3 and RQ2.4. What are the main benefits challenges posed by the digital twin?

The principal benefits and challenges of the implementation of the digital twin in RQ2.3 and RQ2.4 are detailed in Table 4.2, which is partially aligned to *Ch6 (difficulty* *in quantifying the ROI).*

Table 4.2: Benefits and challenges of implementing the digital twin

Benefits	**Challenges**
Process optimisation	Lack of data to train the simulation model
Real time monitoring	Model calibration and synchronisation
Virtual commissioning enhancement	Data model adjustment: only necessary at required level
Reduction of time and cost	Standardisation of interfaces and interoperability
Quality improvement	ROI: effort vs benefit
Remaining Useful Life (RUL) computation	
Viability analysis	

4.2.9 RQ3.1. What are the tests carried out during the commissioning process?

Virtual commissioning verifies and validates the system before deployment to operations by performing a series of verification tasks (RQ2.1). These could include a wide range of tests such as mechanical verification, validation of the PLC, validation of the CNC configuration, CNC part program validation, controller hardware verification, and electrical system verification.

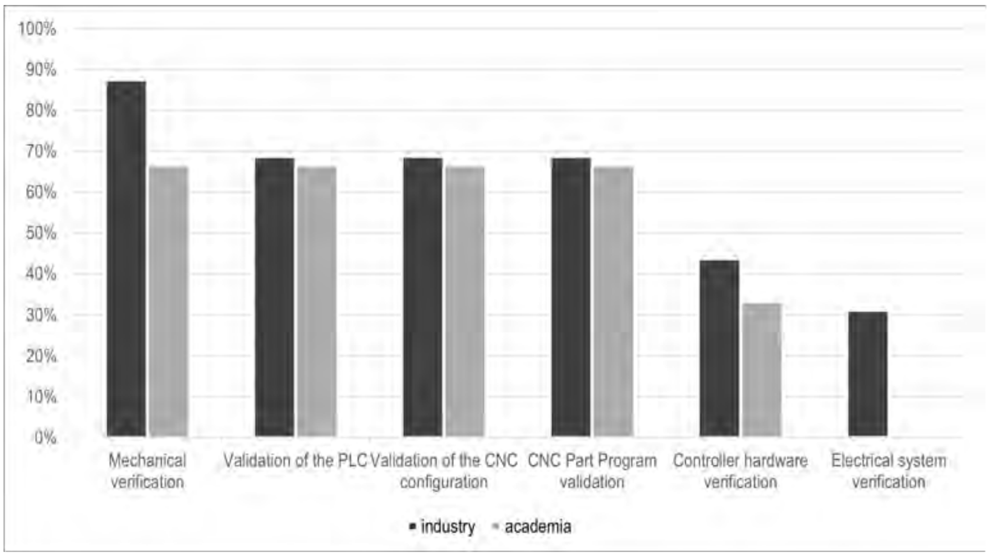


Figure 4.9: Validation and verification tests

Respondents were asked to identify verification and validation tasks performed during virtual commissioning in RQ3.1. The results presented in Figure 4.9 show that mechanical verification tests ranked the highest in industry (88%), followed by validating the PLC, CNC configuration and CNC part program (69%). A further 44% of industry tests the controller hardware, and 31% verifies the electrical system. Academia reported similar results, albeit with a slightly lower percentage. This clearly indicates the importance placed upon validation and verification across all engineering disciplines, thereby justifying the importance of our research focuses.

4.2.10 RQ3.2. Are these tests automated?

RQ3.2 was designed to identify current industrial testing practices for virtual commissioning. Industrial practitioners were asked whether tests were performed manually, with the help of software (semi-automated), or automated (Figure 4.10).

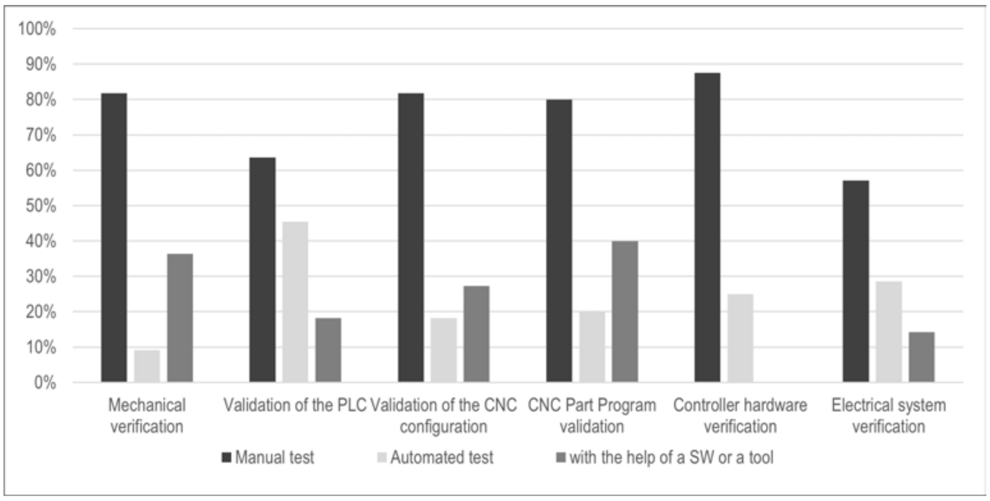


Figure 4.10: Testing procedures

The majority of tests are conducted manually in industry: mechanical verification (82%), validation of the PLC (64%), validation of the CNC configuration (82%), CNC part program validation (80%), controller hardware verification (88%), and electrical system verification (57%). The highest occurrence of automation was PLC Validation, however even in this instance, automated virtual commissioning accounted for only 45% of testing the PLC. A low level of automation was reported in the remaining fields, (29%), and the highest level of semi-automated tests was CNC part program validation (40%). These results are partially linked to *Ch7 (time and effort required to test PLCs)* and *Ch8 (dealing with frequent errors)*. This clearly indicated a lack of uptake of automated testing practices in the industrial sector. As a result, the industrial survey supports the need of *Research Focus #4* identified in the literature review.

4.2.11 RQ3.3. Is there a need for automating tests?

RQ3.3 asks industry and academic practitioners if it is beneficial to automate the tests described in RQ3.1. Figure 4.11 depicts the percentage of those in favour of automating mechanical verification, validation of the PLC, validation of the CNC configuration, CNC part program validation, controller hardware verification, and electrical system verification.

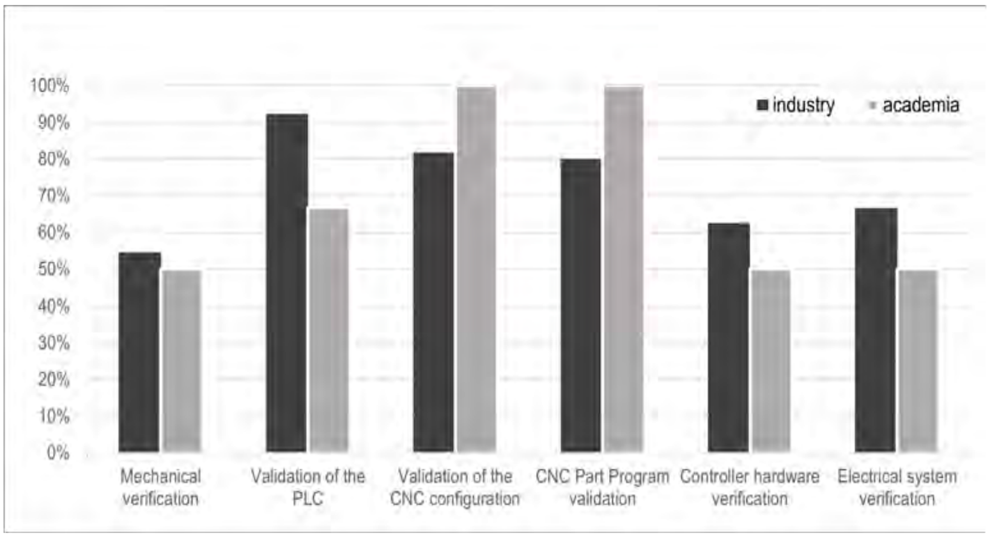


Figure 4.11: Need for automating testing practices

Automating tests for the validation of the CNC configuration, CNC part program validation, and PLC validation scored highly for both academia and industry. The first two are considered critical by academic respondents (100% believe these should be automated), and were also selected by 82% of the industry. Ninety-two% of industry practitioners think automating the validation of the PLC is crucial, in contrast to 67% of academia. Those surveyed in industry also support automating the verification of electrical system (67%), controller hardware verification (63%), and the verification of the mechanical system (55%).

These results underscore the need for improved virtual commissioning testing procedures by applying the aforementioned validation and verification tasks, further reinforcing *Research Focus #4*.

4.2.12 RQ3.4. What are the main testing challenges in the development and operation of a machine tool?

Lastly, the challenges faced by industry in conducting verification and validation tests are outlined in RQ3.4. The survey responses are depicted in Figure 4.12, including, problems related to controller hardware and software, simulator/emulator software errors, PLC related errors, malfunctions in the mechatronic system, communication and integration issues, delays in execution time, lack of precision and stability, unexpected circumstances, and problems arising from earlier development stages.

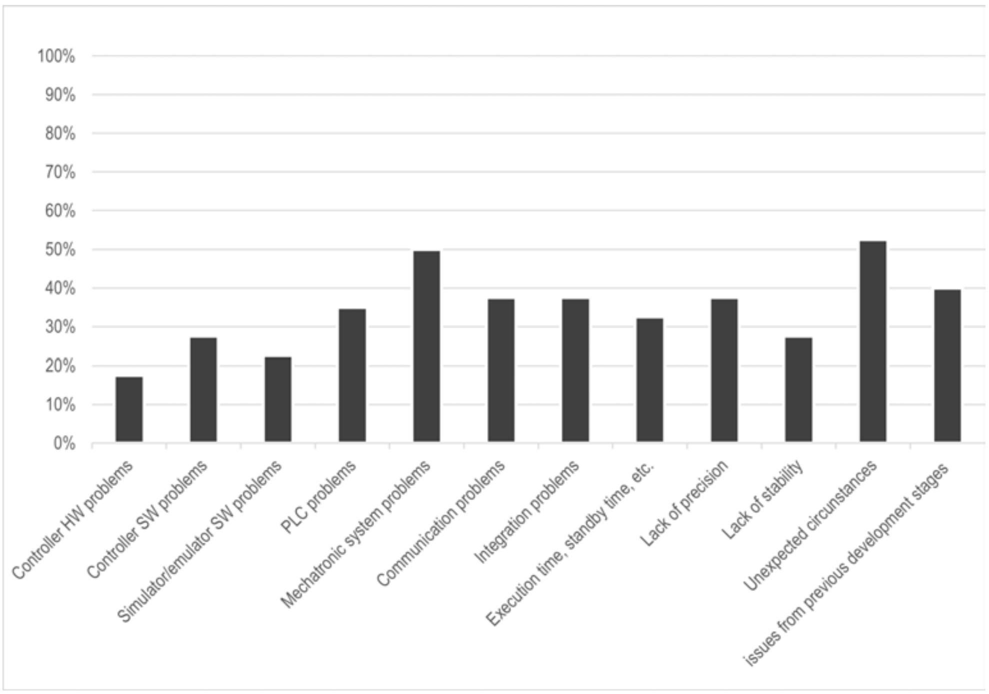


Figure 4.12: Testing challenges

The wide range of challenges inherent in the testing process are highlighted in these findings, with unexpected circumstances and mechatronic system problems ranking as the most common. Among other issues, PLC-related problems also appear to be a significant challenge, further compounding the need for virtual commissioning and testing practices indicated in our research focuses.

4.3 Discussion

The results obtained from our survey have reinforced the challenges claimed in Section 1.1, as well as the research focuses stated in 3.5. The results clearly reveal an evident need for virtual commissioning to reduce time to market and commissioning errors (*Ch4: time to market and costs*). It was encouraging to note that academic respondents, for the most part, identified the potential of carrying out verification activities across various engineering disciplines during the entire development process. Nonetheless, the advantages of virtual commissioning remain under appreciated in industry, and it is still relegated to the last stage of the development process (*Ch4: time to market and costs*). Hence, there exists a need to encourage industrial companies to conduct virtual commissioning throughout the whole development process (design, engineering, commissioning) to reduce time to market, errors and costs.

There are several challenges to implementing digital twin and virtual commissioning in industry, in relation to *Research Focus #1*. One important consideration is that the efforts required to invest in such technologies do not always yield the required ROI (*Ch6: difficulty in quantifying the ROI*). Other concerns include standardisation and interoperability (*Ch3: interoperability issues*), software limitations, lack of available data and model adjustment, and accuracy of simulations. A traditional mindset in industry was also flagged as a barrier to investing in such technologies.

The fact that the major part of testing practices are still conducted manually in industry underpins *Ch7 (time and effort required to test PLCs)*, and *Ch8 (dealing with frequent errors)*. This demonstrates an obvious need for automation, reinforcing the need of *Research Focus #4*. In this regard, industry respondents did identify the benefits of improving testing practices through automation. Accordingly, verification and validation should be automated in the near future, to expedite virtual commissioning practices and strengthen the competitive edge of industrial manufacturers.

4.4 Threats to Validity

4.4.1 Internal Validity

An internal validity threat is the sample size of the survey. This was mitigated in the present study by selecting a wide range of industrial and academic participants with diverse backgrounds (mechanics, electronics, automation, telecoms, computer science, ICTs, etc.) and years of experience. The complexity and understandability of the questions could also be considered an internal threat to validity. We provided simple definitions of key vocabulary at the start of the questionnaire to reduce this threat. Those surveyed were also given the option to make free comments on open questions about specific topics.

4.4.2 External Validity

The demographic makeup of the samples might be considered an external threat. This concerns the participating companies, which were mostly restricted to the Basque Country region in northern Spain. For this reason, it was not possible to generalise the results. Rather, the findings can be considered representative of the machine tool sector of that area, which accounts for the great majority of machine tool production in Spain.

4.5 Conclusions

This chapter has presented the results of an empirical survey carried out to understand existing needs and challenges in adopting virtual commissioning and the digital twin, in line with our *Research Focus #1*. We took as our focus the machine tool sector and research centres, located mainly in the Basque Country region of northern Spain.

The survey was designed to benchmark industry against academia and to gain greater understanding of industrial needs and challenges in virtual commissioning practices. Participants were asked to respond to three sets of research questions related to: virtual commissioning practices, the technological readiness of the digital twin for virtual commissioning, and testing procedures and practices when commissioning.

Overall, the findings show that industry lags behind academia. While the literature on digital twin-based virtual commissioning is growing swiftly, the practical implementation of this technology is not yet appropriately addressed. As a result, there remain several testing challenges that need to be resolved in industry.

The survey reveals a number of requirements that are essential for success in industry:

- Increasing awareness of the benefits of virtual commissioning (addressing *Ch6: difficulty in quantifying the ROI*).
- Conducting virtual commissioning early in the development process (addressing *Ch4: time to market and costs*).
- Promoting interoperability and standardisation (addressing *Ch3: interoperability issues*).
- Utilising high fidelity simulations (addressing partially *Ch5: lack of collaborative and holistic solutions*).
- Shifting away from the traditional mindset prevalent in industry (addressing *Ch6: difficulty in quantifying the ROI*).

- Implementing a holistic multi-domain solution (addressing *Ch5: lack of collaborative and holistic solutions*).
- Adopting automated testing practices (addressing *Ch7: time and effort required to test PLCs* and *Ch8: dealing with frequent errors*), as stated in *Research Focus #4*.
- In overall, addressing the gap between research and industry (*Ch9: lack of knowledge transfer*).

In the following chapters, therefore, we introduce a theoretical framework for virtual commissioning (*Research Focus #1*), and present an automated testing methodology for industrial PLCs (*Research Focus #4*) that incorporates software engineering practices for PLC testing (*Research Focuses #2-3*).

Chapter 5

Theoretical Framework

Contents

In this chapter, we present a theoretical overview of the dissertation. Four research objectives (Section 5.1) are defined, together with the hypotheses (Section 5.2). A theoretical framework for testing industrial PLC software in a cost-effective manner is also proposed (Section 5.3). We then explain the case studies that were used to validate the effectiveness of the solutions proposed in the theoretical framework (Section 5.4). Lastly (Section 5.5), the case studies employed for validating each contribution (Section 1.3) are outlined.

5.1 Research Objectives

In a volatile global marketplace, the manufacturing industry must respond rapidly and with agility to constantly changing customer needs. The PLC is the main controller of manufacturing systems, and thus PLC programs require frequent reconfiguration to address new process requests. Hence, it is crucial that modifications to the PLC code are thoroughly tested to eliminate errors, particularly when working with safety critical systems.

At present, PLC testing is predominantly performed manually, which is time-consuming and requires considerable human effort (see Section 3.5). Moreover, manual testing is error-prone, which increases costs due to unexpected failures and delays during the commissioning process. These disadvantages are further compounded when testing highly reconfigurable PLC programs. In this regard, some authors have focused on automating the testing process of these programs by generating automated test cases. However, the cost-effectiveness of these tests is yet to be optimised, and their transfer to industrial PLCs remains a significant challenge.

Hence, the main goal of this research work is to:

Provide a methodology to cost-effectively test and commission highly reconfigurable industrial PLC programs.

To achieve this end, the first objective *(Objective 1)* is to build an interoperable framework that enables the continuous commissioning of PLCs (derived from *Research Focus #1*). Three technical objectives are also defined to cost-effectively test and commission PLC programs within this framework:

- *Objective 2: develop a tool to automatically generate and evaluate test cases for a wide range of IEC 61131-3 standardised FBD programs (derived from Research Focus #2).*

- *Objective 3: develop and evaluate test case selection metrics and algorithms for highly reconfigurable PLCs (derived from Research Focus #3).*
- *Objective 4: develop and evaluate a methodology to automatically perform SiL-based commissioning on industrial PLCs (derived from Research Focus #4).*

Table 5.1 summarises the relationship between the identified gaps, research focuses, and objectives:

Table 5.1: Relationship between objectives, research focuses and gaps

Objectives	**Research focuses**	**Gaps**
1	1	1, 2, 3, 4
2	2	7
3	3	8, 10
4	4	9

Each of the objectives defined here is directly linked to the following technical contributions:

- FBDTester 3.0: coverage-based automated test generation and execution tool for IEC 61131-3 FBD programs. This module extends on a previous work [SJB18] to automatically create coverage-based tests and evaluate test cases. This new version uses a new solver to solve non-linear arithmetic functions defined in the IEC 61131-3 standard (*Objective 2*). It also includes a new functionality to execute and obtain cost-effective data measurements, as well as test oracles of the subject tests (*Objectives 2-4*).
- MuFBDTester 3.0: mutation-based automated test generation tool for IEC 61131-3 FBD programs. This module is also based on [SJB18] to automatically generate mutation-based tests for the vast majority of IEC 61131-3 function groups (*Objective 2*).
- A multi-objective search-based test selection approach to optimise the selection of tests in a cost-effective manner (*Objective 3*).
- A methodology to automatically test Siemens-specific PLC programs in TIA portal using test oracles (*Objective 1, Objective 4*).

5.2 Research Hypotheses

Based on the above objectives, the following research hypotheses were defined:

- *Hypothesis 1:* coverage-based and mutation-based testing enables the automated generation of test cases for most of the IEC 61131-3 function groups, including complex and non-linear arithmetic functions. This hypothesis corresponds to research Objective 2.

- *Hypothesis 2:* search-based test selection algorithms optimise the selection of the generated test cases in a cost-effective manner. This hypothesis corresponds to research Objective 3.

- *Hypothesis 3:* the automated PLC testing methodology can test highly reconfigurable industrial PLCs with the use of test oracles. This hypothesis corresponds to research Objectives 1-4.

5.3 Overview of the Theoretical Framework

The digital twin is an agile framework that greatly enhances the capacity of manufacturing systems to meet frequently changing market needs. Hence, we introduce the digital twin as the enabler of DevOps across the product life cycle of a shop floor [QES20], as described in Figure 5.1. In this case, the digital twin technology reduces the gap between development and operations as it seamlessly interchanges data throughout the entire life cycle.

In this way, new customer requirements can be addressed on the go, by providing a proactive and continuous optimisation process. This continuous production system could make the shift to virtual commissioning, and reduce development costs and time to market.

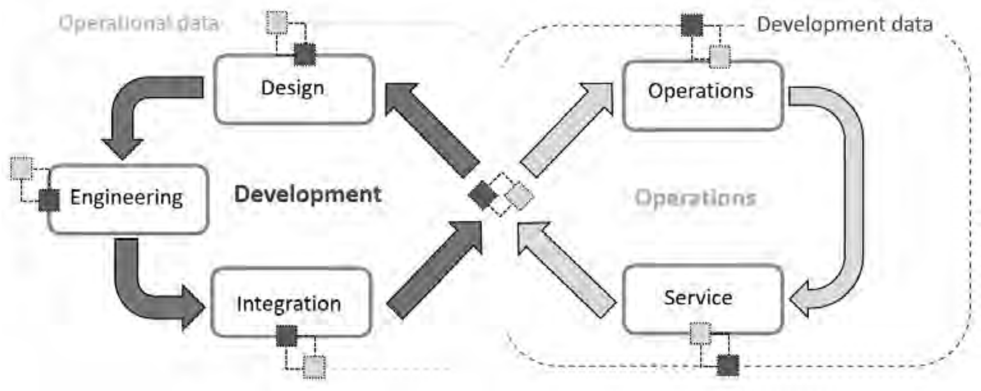


Figure 5.1: Digital twin-enabled DevOps approach [QES20]

Our methodology is implemented in the integration stage of the PLC development process, in which we propose a mechanism to cost-effectively validate the logic of PLC programs for subsequent SiL testing, as illustrated in Figure 5.2.

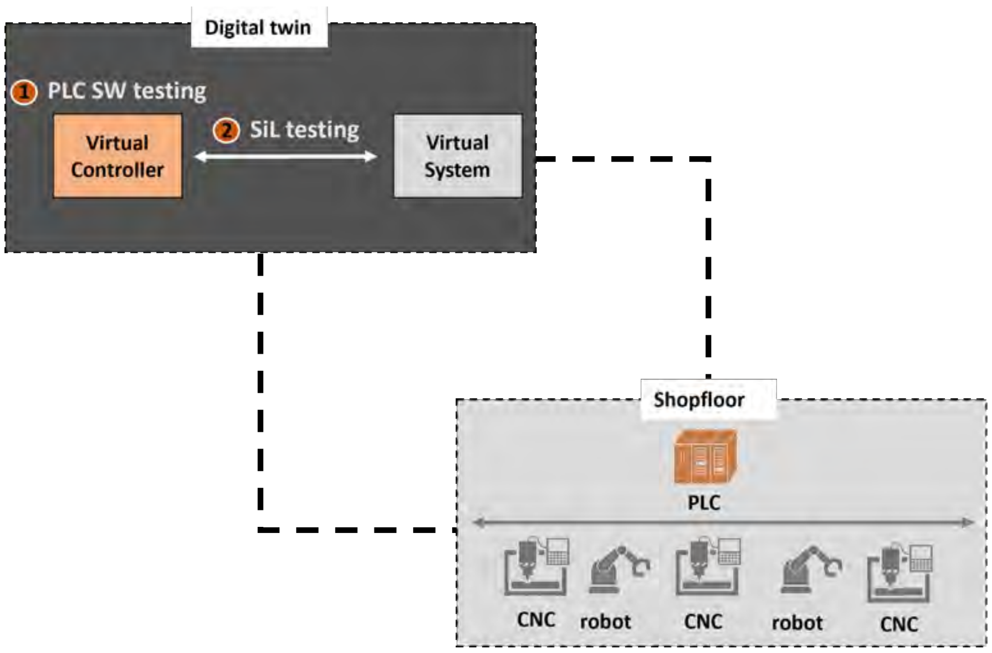


Figure 5.2: Digital twin-based continuous testing of PLCs

Digital twins are well positioned to enhance virtual commissioning practices, as they can address synchronisation issues and misalignments between virtual representations and the physical system [QES20].

Despite the wide variety of tools and technologies commercially available, no existing solution integrates multiple domain simulation and emulation technologies into a single and unified platform. As a result, companies are reluctant to invest in simulation technologies for virtual commissioning, because the efforts required lack justification in terms of ROI. This poses a serious obstacle to the adoption of virtual solution strategies.

To this end, we present a digital twin-based theoretical framework that could be used to enhance virtual commissioning practices in the manufacturing sector. We defined a four-layer digital twin architecture, in line with the digital twin-based virtual commissioning concept of Shen et al. [TLHN20].

The physical layer consists of two different automation levels, i.e. machine level and cell/production line level, as illustrated in Figure 5.3.

1. **Machine level**: an individual machine tool that comprises a single CNC, PLC

and/or robot.

2. **Cell/Production level**: multiple machine tools together with multiple controllers (i.e. CNC > 1, PLC > 1).

The cyber layer comprises the digital twin of the physical devices, coupling multiple domain models. These are made up of a wide range of vendor-specific controllers and simulation technologies.

The principal challenge of a virtual commissioning system is the integration of a broad set of vendor-specific controllers and simulation technologies into a single environment. For this reason, an interoperable gateway is employed as a middleware to seamlessly integrate all these devices into a collaborative platform. The middleware in this case should not only provide communication exchange among the devices, but also offer a mechanism to retrieve operational data and define functions or specific tasks and services. This follows the 5-dimensional digital twin architecture defined by Tao et al. [TZN19].

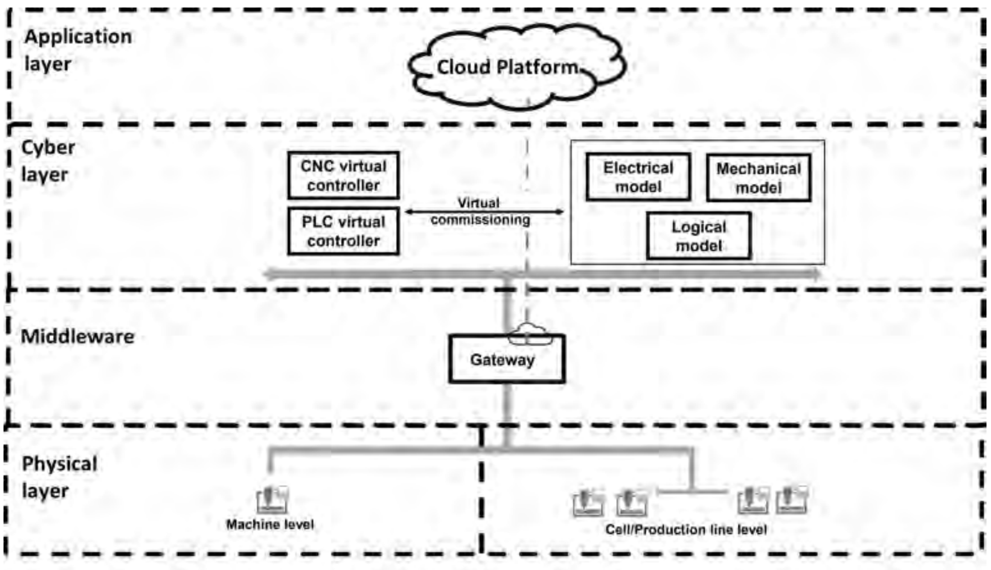


Figure 5.3: Digital twin-based virtual commissioning architecture – based on [TLHN20]

Our approach, however, does not rely on the application layer and is primarily limited to the cyber layer. We utilised an offline PLC Virtual Controller (Digital Model) to simulate the PLC Controller. Nevertheless, the proposed methodology and process should still hold valid with a fully synchronised PLC Controller.

Figure 5.4 illustrates our methodology to cost-effectively test the software of highly reconfigurable PLCs in industry. The methodology was implemented in the

Siemens TIA Portal environment, which is designed to program, simulate, and test PLC programs. A SiL approach is proposed to validate the changes before commissioning and deploying the new PLC code into operations. At the time of writing, not all systems were in place, and our case studies were conducted in an offline simulation environment. In future works, however, we envisage a fully online digital twin-based PLC commissioning solution, based on the present simulation-based testing approach.

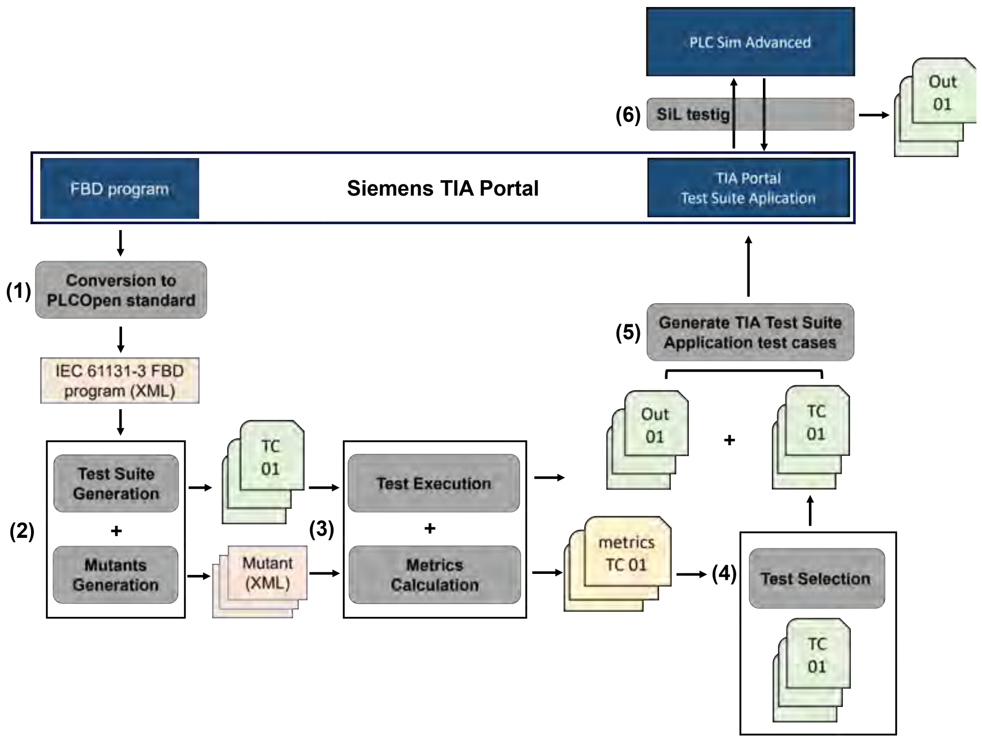


Figure 5.4: Overview of the methodology

We designed a vendor-agnostic and interoperable solution (*Objective 4*) applicable to a wide range of IEC61131-3 standardised function groups, including non-arithmetic functions (*Objective 2).*

Our test solution is based on the FBDTester 2.0 and MuFBDTester developed by Korea Advanced Institute of Science and Technology (KAIST) university [SJB18], which generate coverage-based and mutation-based test cases for PLCopen XML PLC programs. These tools employ data-path flow testing techniques, thereby eliminating the need to perform any additional model-to-model transformation (*Objective 2).*

Given that time plays a critical role when testing, we present a multi-objective test case selection approach based on a set of adequacy criteria (*Objective 3*). This technique is designed to maximise the detection of faults, while minimising commissioning time. To this end, we empirically evaluated the effectiveness of the test selection approach in detecting implementation errors.

As a proof of concept, the proposed approach was implemented in Siemens TIA Portal Test Suite Advanced (*Objective 1, Objective 4*). In this way, the newly modified PLC code is validated against PLCSim Advanced, which is used to simulate the PLC controller. This is of particular interest for SiL testing, as the logic of the controller can be safely tested without the need for physical hardware.

Our approach enhances PLC testing practices – a manual and error-prone process – by transferring novel testing practices to industry. This cost-effective testing methodology is expected to significantly reduce implementation errors while shortening commissioning time.

The methodology presented in Figure 5.4 comprises the following steps, which are linked to their respective Objectives and Chapters in Table 5.2.

1. As the developed solution is designed for PLCopen XML programs, Siemens PLC programs should first be converted to PLCopen XML standards.

2. Next, the test generation tools parse the PLCopen XML code to define IEC 61131-3 FCs, FBs, constant values, and input and output signals. This information is used to define the data paths and data path conditions for the FBD. The test generation tools then generate test cases based on structural coverage and mutation criteria.

3. The cost-effective metrics (including coverage, calculation, execution time and fault detection capability) of the generated test cases are evaluated then in the FBDTester 3.0.

4. These measures are used as fitness metrics to select the optimal subset of the test suite for error detection.

5. The execution results of the selected test cases are employed as oracles to define test output assertions. With this information, TIA Test Suite application tests can be generated, in which we execute the PLC code with given input data in the simulation environment.

6. Finally, TIA Portal establishes a connection with PLCSim Advanced simulation environment and compares the obtained results with the expected behaviour. In this way, the newly modified PLC code is validated through the imported application test cases in TIA Portal Test Suite.

Table 5.2: Technical contributions and their correspondence to the aforementioned steps, objectives, and chapters of the document

Technical Contributions	Steps	Objectives	Chapters
Technical Contribution 1: FBDTester 3.0	2, 3	2, 3	6, 8
Technical Contribution 1: MuFBDTester 3.0	2	2	6, 8
Technical Contribution 2: Multi-objective search-based test selection approach	4	3	7
Technical Contribution 3: Automated PLC testing approach for TIA Portal	1-6	1, 4	8

5.4 Case Studies

The methodology was developed under the framework of the DiManD ITN programme. The DiManD project supports the development of a holistic framework for future highly intelligent, adaptable, and responsive manufacturing infrastructure. In this context, the main objective of this body of research was to build an agile and robust testing methodology to validate systems undergoing frequent changes. Two industrial case studies were defined to validate the proposed methodology:

1. Omnifactory, which is the future automated aerospace assembly demonstrator at UNOTT, and is envisaged to showcase the DiManD integrated project as one of the main beneficiaries of the project.

2. A CNC machine tool solution at Danobatgroup, which is an industrial partner of the DiManD project.

In addition, four use cases from the Korea Nuclear Instrumentation and Control System (KNICS) reactor protection system were employed to validate the developed test generation and test the optimisation techniques.

5.4.1 Omnifactory

The Omnifactory will showcase a national experimental testbed and technology demonstrator for smart manufacturing systems in the aerospace, space and defence, and automotive sectors[STS+20]. It builds on previous research into the Future Automated Aerospace Assembly Demonstrator (FA3D) and Evolvable Assembly Systems (EAS) project demonstrators [STS+19]. A flexible and reconfigurable assembly demonstrator that is transformable and scalable is envisioned. The platform includes a reconfigurable floor, robotic automation platforms, precision metrology systems, and digital and manufacturing technologies, as depicted in Figure 5.5.

The Omnifactory consists of a mix of both automated and manual assembly solutions to quickly respond to any customer needs. In particular, it is designed to assemble multiple product families (i.e. fuselage, wing, nacelle), work at different

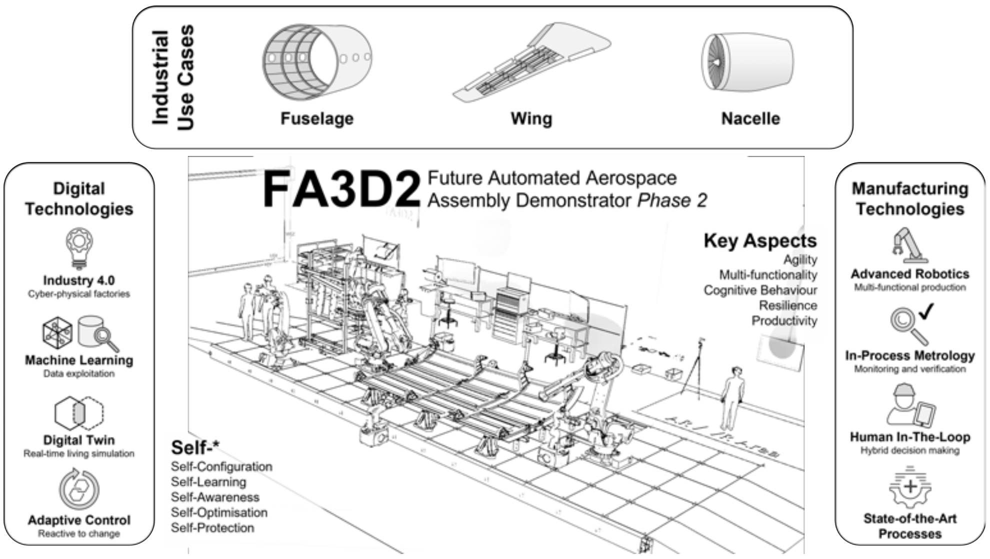


Figure 5.5: Omnifactory overview [STS+20]

production rates, and manage ever-changing processes based on demand. As such, four generic use cases are defined in the testbed:

- Integration of a new end-effector.
- Integration of new automation platform (robot, etc.).
- Whole-cell reconfiguration (in terms of physical layout and/or process).
- Business as usual (the cell under operation).

All of these changes and reconfigurations are likely to require modifications in the system layout, and consequently in the PLC controller. To ensure operational safety, every change must be tested before being deployed to operations, ensuring it is error-free. Figure 5.6 details the main steps involved in the process of automatically updating the system when a new product or process request is received. The contribution of the present research tackles specifically Steps 4-5.

The use case was retrieved from the FA3D – the previous automated assembly platform of Omnifactory – since the PLC of the Omnifactory was not yet available. Omnifactory was scheduled to include multiple robots and machine tools in the layout, therefore the drilling FBD used in the FA3D could still be valid to control the spindle and drill slide. As a proof of concept, two FBD networks were selected from the Drilling use case. The characteristics are defined in Table 5.3:

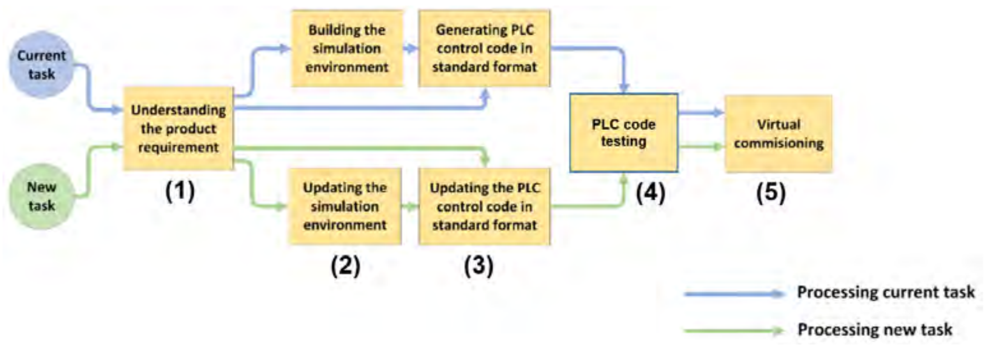

Current task
New task
Understanding the product requirement
(1)
Building the simulation environment
Generating PLC control code in standard format
Updating the simulation environment
Updating the PLC control code in standard format
(2)
(3)
PLC code testing
(4)
Virtual commisioning
(5)
Processing current task
Processing new task

Figure 5.6: Omnifactory system reconfiguration framework

Table 5.3: Omnifactory use case size: number of FBD blocks, inputs, and outputs

Use case	#blocks	#inputs	#outputs
Drilling FBD Network 1	11	12	5
Drilling FBD Network 2	11	19	1

5.4.2 Danobatgroup

Danobatgroup is one of the largest European machine tool builders, and IDEKO is an industry-driven technology research centre of Danobatgroup. The secondment carried out at IDEKO was focused on testing FBD programs of Danobatgroup machine tools.

In the second case study, a means was provided to test FBD programs in TIA Portal, by generating TIA Portal Test Suite-specific test cases. Furthermore, the developed test selection approach ensures cost-effective testing of PLC programs in the simulation environment (i.e. PLCSim Advanced) for commissioning purposes. This has considerable impact on the PLC development process of industrial machine tool controllers at Danobatgroup. The deployment process of PLCs for new machine tools is illustrated in Figure 5.7.

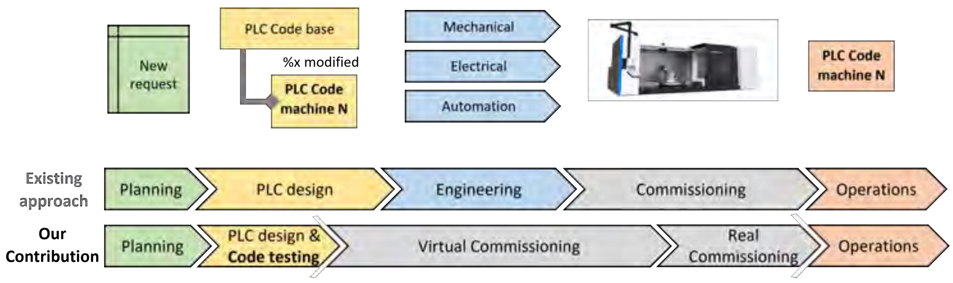


Figure 5.7: PLC development process at Danobatgroup

Every time there is a new machine-tool request, the PLC program is generated by making some changes to the existing PLC base code. On average, 5% of the PLC base code is amended to tailor the PLC program to the requirements of the new machine. Currently, the newly modified code is not tested until the commissioning stage which frequently results in errors due to unexpected behaviours. However, it is difficult to pinpoint the source of the error, as commissioning involves other parts of the system as well, including the mechanical system, electrical system, etc. The entire system – including the PLC code – needs to undergo rigorous testing, and on occasion be redesigned, until it is fully accepted during the Site Acceptance Testing (SAT).

This process is error-prone, as the system is not tested until all physical parts are in place. Commissioning is therefore costly, and causes severe delays to the deployment of new machine tools and their respective PLCs. Hence, the need to optimise the process.

Specifically, three use cases were employed – a Robot FBD, Safety FBD, and Gantry FBD – to test the Danobatgroup case study, in which the main characteristics are detailed in Table 5.4:

Table 5.4: Danobatgroup use cases size: number of FBD blocks, inputs, and outputs

Use case	**#blocks**	**#inputs**	**#outputs**
Safety FBD	5	10	1
Gantry FBD	17	26	8
Robot FBD	4	10	1

5.4.3 KAIST Use Cases

For testing purposes, we employed use cases widely used by KAIST university [SJB16, SJB18, JSC$^+$14] to test IEC 61131-3 defined FBD programs. Most of these use cases are parts of the PLC code of a Bistable Processor (BP) of the reactor protection system developed by KNICS [ind06].

The BP is considered a safety critical system, which must undergo rigorous testing in accordance with government regulation. In total, there are about 20 different FBD modules. In our study, we selected three modules from the KNICS project (i.e. simTRIP, FFTD, and FRTD) and two additional modules utilised at KAIST [SJB16, SJB18], namely LAUNCHER, and simGRAVEL. The latter two were selected as complementary modules since they include IEC 61131-3 function groups not covered at KNICS.

simTRIP is a simplied module of the BP used in the KNICS project. It consists of a GE (Greater than or Equal to) comparison FC, an AND logical FC, and a TON (ON-delay Timer) FB, as indicated in Figure 5.8. In the TON case, the output will turn on after a Pulse Time (PT) delay. In this case, the TON output Q is updated

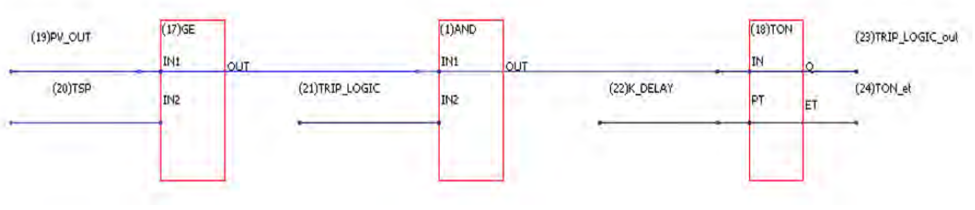


Figure 5.8: simTRIP FBD module

to true when the input IN has been true for the duration of a PT. The output ET represents the elapsed time that the input IN has been true. Specifically, if the process value (PV_OUT) is greater than or equal to Trip Set Point (TSP) and TRIP_LOGIC is not true for K_DELAY time, the output TRIP_LOGIC is set to true. As a result, TRIP_LOGIC sends a trip signal to shut down the nuclear reactor.

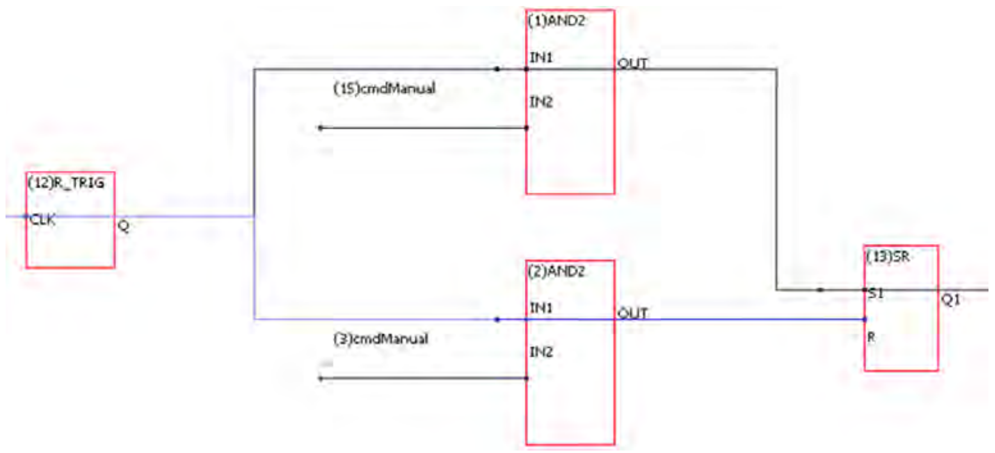


Figure 5.9: LAUNCHER FBD module

The LAUNCHER use case comprises a rising edge trigger (R_TRIG) FB from the edge detection group, AND operators from the logic function group, and a Set-Reset (SR) FB from the bistable elements group (Figure 5.9).

The use case simGRAVEL (Figure 5.10) consists of an AND logic FC, a TON block from the timers group, and a CTU count up FB from the counters group.

Lastly, FFTD and FRTD are larger use cases that include some additional IEC 61131-3 function groups, such as comparison FCs, linear arithmetic FCs, and selection FCs.

Table 5.5 sets out the size of the use cases, including the number of FCs or FBs, the number of input variables, and the number of outputs.

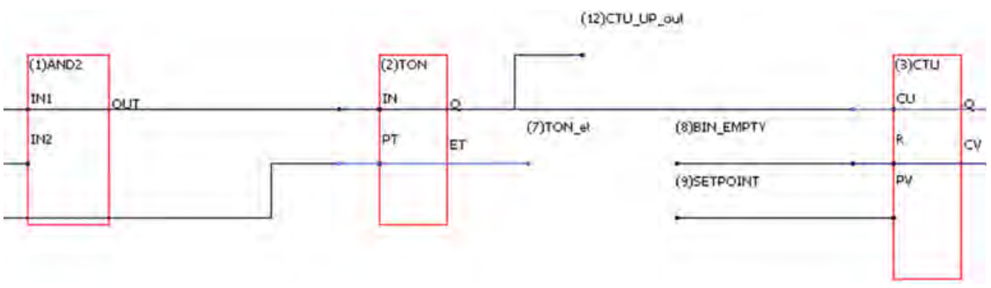


Figure 5.10: simGRAVEL FBD module

Table 5.5: KAIST use cases size: number of FBD blocks, inputs, and outputs

Use case	**#blocks**	**#inputs**	**#outputs**
simTRIP	3	4	2
LAUNCHER	4	2	1
simGRAVEL	3	3	4
FFTD	29	12	8
FRTD	29	12	8

5.5 Case Studies Employed for Validating each Contribution

Each of the contributions was independently validated with one or more case studies, as detailed in Table 5.6. The KAIST use cases were used for validating the FBDTester 3.0, MuFBDTester 3.0 and the multi-objective search-based test selection approach, in which the simTRIP use case was also utilised to complement the validation of the automated PLC testing approach for TIA Portal. Besides, the latter contribution was validated with the presented two industrial case studies, i.e. the Omnifactory and Danobatgroup, which use Siemens PLC programming environments.

Table 5.6: Case studies used to validate each of the contributions

Contributions	**Omnifactory**	**Danobatgroup**	**KAIST**
FBDTester 3.0			X
MuFBDTester 3.0			X
Multi-objective search-based test selection approach			X
Automated PLC testing approach for TIA Portal	X	X	X

Part II

Test Generation

Chapter 6

FBD Test Generation

Contents

6.1 Introduction

In this chapter, we present a test generation and evaluation approach to automatically create coverage-based, mutation-based, and random test cases to test IEC 61131-3 FBD programs. An automated evaluation process to calculate cost-effective metrics and derive output oracles is also proposed. This chapter describes our first technical contribution, referred to as *Technical contribution 1* in Section 1.3, which is broken down into three original sub-contributions:

- **Sub-contribution 1:** A new SMT solver to generate test cases for IEC 61131-3 FBD programs, including non-linear arithmetic functions.
- **Sub-contribution 2:** Calculation of cost-effective metrics to assess and optimise the selection of test cases. These test cases and metrics will be further used in Chapter 7.
- **Sub-contribution 3:** Automated test execution, which will serve as test oracles during SiL testing that will be described in Chapter 8.

The presented sub-contributions are directly linked to the contributions, objectives, and hypotheses described in Table 8.9:

Table 6.1: Chapter 6 contributions, objectives and hypotheses. Partially related ones are marked with an asterisk (*) symbol

Sub-contributions	**Contributions**	**Objectives**	**Hypotheses**
1	Technical Contribution 1	2	1
2	Technical Contribution 1	2, 3*	2*
3	Technical Contribution 1	2, 4*	3*

To this end, the main objective of this chapter is stated below:

Objective 2: Develop a tool to automatically generate and evaluate test cases for a wide range of IEC 61131-3 standardised FBD programs.

The chapter is structured as follows: Section 6.2 outlines the proposed methodology, and the test generation process is detailed in Section 6.3 (Sub-contribution 1). Test oracles and cost-effective metric calculations (Sub-contributions 2-3) are described in Section 6.4. The approach is evaluated in Section 6.5 and the conclusions are presented in Section 6.6.

6.2 Overview of the Approach

In this section, we outline a novel approach to automatically generate test data for a wide range of FBD programs. This method employs a combination of techniques to create input data that can be used to test the functionality of FBD programs. The goal of this approach is to improve the efficiency and effectiveness of FBD program testing. This is achieved by reducing the need for manual testing while increasing the fault detection capability. The approach focuses on Steps 2-3 of the methodology presented in Figure 5.4, which is further detailed in Figure 6.1, and is divided into 4 main tasks.

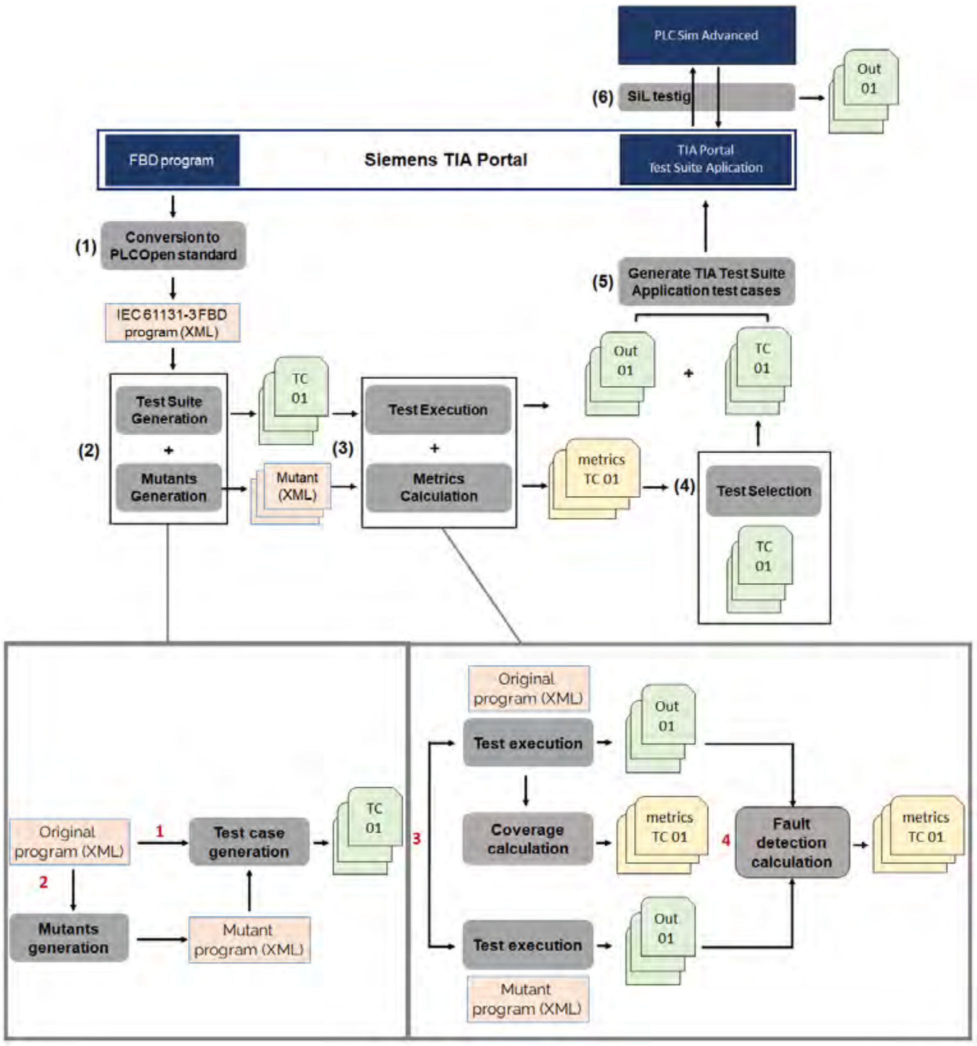


Figure 6.1: Test data generation overall approach

In the first task, test cases are generated automatically based on specific criteria.

These criteria include coverage-based, mutation-based, and random approaches. The standardised IEC 61131-3 PLC program is used as input, and test generation requirements are produced based on the data flow characteristics of the subject program and the selected criteria. For example, if the coverage-based criteria is selected, the approach will aim to generate test cases that cover as many parts of the program as possible. In the case of mutation-based criteria, the focus will be on maximising the detection of possible faults (i.e. mutants) in the program. Finally, test cases are also generated randomly, which can be useful for identifying unexpected behaviours in the program.

In the second task, faults are inserted into the original FBD program to create faulty versions. In the absence of information about real faults, faults are artificially generated with different sets of mutants, which has been demonstrated to be an appropriate substitute [JJI$^+$14]. The faulty programs are then used in tasks 3 and 4 to determine the capability of the program to detect errors, thereby measuring the effectiveness of the test cases generated in the first task.

In the third task, the test cases are executed in the original program to retrieve cost-effectiveness metrics. This includes calculating the coverage of the test cases, measuring the execution time of the tests, and calculating the outputs. These outputs will serve as oracles to compare the expected and real behaviour of the program. The test cases are once again executed in the generated mutant programs to derive the fault detection capability in task 4.

We utilised existing tools in this study, such as the FBDTester 2.0 [SJB18] and the MuFBDTester [LJB22b] as a foundation. We then enhanced these tools by incorporating a new solver Yices 2 SMT2 [Dut14] to manage the majority of function blocks, including those with non-linear arithmetic operations. All necessary modifications were also made to support the third task of the approach. The result is newly developed versions of these tools, namely FBDTester 3.0 and MuFBDTester 3.0, which were tailored to meet the requirements of this study. In particular, FBDTester 3.0 was used in tasks 1 and 3 in conjunction with the new solver Yices 2 SMT2. We employed the MuFBDTester 3.0 and the Yices 2 SMT2 solver in tasks 1 and 2 to generate mutation-based test cases and faulty programs. Lastly, an in-house script was developed in Java (named *ResultsComparer.java*) for the final task.

6.3 Test Case Generation Process

This section sets out the test case generation process for testing FBD programs. Test cases are defined with various criteria approaches, namely coverage-based, mutation-based, and random. The process follows the steps described in Figure 6.2.

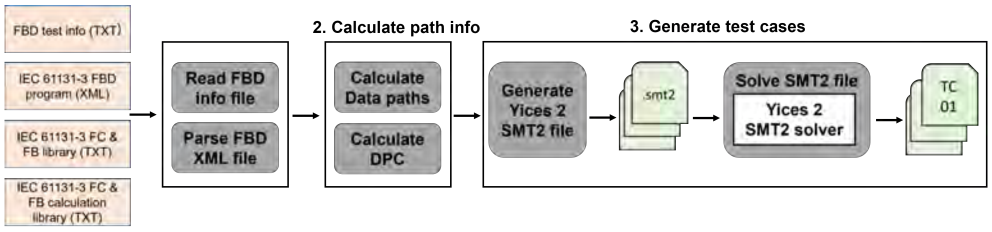


Figure 6.2: Test case generation process

First of all, the IEC 61131-3 FBD program and the respective test file are read and parsed to load the PLC data. The test file contains data about the inputs, outputs, and constants of the FBD program. The FBD contains the Program Organisation Unit (POU) of the program code, which details all connections, FCs and FBs. This information is then used to calculate the data paths and DPC based on the definitions of the IEC 61131-3 library. In the final step, the Yices 2 SMT2 with the definitions of the data path coverage requirements were generated, which derived the test cases. This was performed with an external SMT2 solver, which solved the specified requirements in the SMT2 file and returned a sequence of feasible test cases.

In the following subsections, the solver and the criteria employed for generating the tests are described in detail.

6.3.1 SMT Solver

The solver is a critical component of the approach, as it generates the test input data to test FBDs based on the specified requirements.

Yices 1 Solver

The FBDTester 2.0 and MuFBDTester tools use the Yices 1 solver to generate test data. This is a SMT solver that decides the validity or satisfiability of logical formulas. The Yices solver generalises the satisfiability problem to complex formulas in first-order theories. This means that the solver takes as input the requirements specified by the chosen criteria, and attempts to solve them as complex mathematical formulas. The solutions derived from this process are the test input data or test cases.

The Yices solver is a powerful tool for generating test data, however, the first version has some limitations when handling complex numerical functions. By way of example, it does not effectively support non-linear arithmetic, and non-constant division operators, which can be present in some FBD programs. This limits its applicability when testing FBD programs that contain such FCs and FBs. Therefore, in this study we integrated the latest version of Yices solver (i.e. Yices 2) to service a wider range of FBDs.

Yices 2 Solver

Yices 2 is the latest version of the Yices SMT solver. It features a number of improvements and new features, including support for non-linear arithmetic formulas.

The Yices 2 solver includes a Model Construction Satisfiability algorithm (MCSAT) to support non-linear arithmetic operators. This is accomplished by combining heuristics, and search-and-resolve loops to construct models of non-linear arithmetic constraints [Jov17].

MCSAT can be used in Yices 2 by calling the *check-sat-using-mcsat function* instead of the *check-sat function*. This former takes a set of constraints as input and returns a model if the constraints are satisfiable. However, MCSAT is not able to return the unsatisfied assertions (unsat core) that led to the unsatisfied result, data which is needed for coverage-based testing in the FBDTester 2.0. This information can be retrieved by using either the Yices API with any of its bindings (e.g. programming languages such as Java, Python, OCaml, Go) or the SMT2 language instead of the Yices native language. Table 6.2 summarises the features supported by each version of Yices for non-linear arithmetic solutions and retrieval of unsatisfied assertions.

Table 6.2: Yices solver characteristics

Tool	Language	Non-linear arithmetic	Unsatisfied assertions
Yices 1	Yices	No	Yes
Yices 2	Yices	Yes	No
Yices 2	SMT2	Yes	Yes

Hence, our approach incorporates the Yices 2 solver with the SMT2 language, as it offers support for both non-linear arithmetic and retrieval of the set of assumptions that caused the unsatisfied result.

6.3.2 Generating the Yices 2 SMT2 File

In the following subsections, the key steps in the process of generating Yices 2 SMT2 files for coverage-based and mutation-based testing are detailed. The first step was to specify the environment and the logic used in the FBD program, together with the constants and input variables. FCs, FBs, DPCs, and data path variables were then defined based on the data flow characteristics of the FBD. Next, coverage-based requirements were established in the data path, and the solver checked the satisfiability of the requirements. If the solution is satisfiable, the input values that led to that solution are retrieved, giving rise to the test case.

Specifying the Environment

Unlike the Yices native language, SMT2 requires more detailed information about the logic and theory to be used when analysing the formulas and requirements. In this case, we selected QF_NIRA as the main logic, which stands for “Quantifier-Free Non-linear mixed Integer-Real Arithmetic”. It supports the combination of integers, real variables, and non-linear arithmetic operations without the need for quantifiers. The options *produce-models* and *produce-unsat-model-interpolants* were set to true to retrieve the values of the unsatisfied submodels, as shown in Code Block 1.

Code block 1 Specifying the environment

```
(set-option: produce-unsat-model-interpolants true)
(set-option: produce-models true)
(set-logic QF_NIRA)
```

Defining Constants

Constants are variables that have a fixed value, defined with the command *define-const* in the SMT2 language. Hence, we defined all constants following the syntax (define-const <name> <type><value>), where <name> is the name of the constant, <type> indicates the type of the variable (i.e. Boolean, Real), and <value> is the value assigned to the constant. As an example, Code Block 2 gives the definition of two constants of type Real: SCAN_TIME and K_DELAY. The former represents the PLC cycle time in milliseconds, and is a fixed value of 50ms. The latter is a programmed time constant used in PLC timers (i.e. TON, TOF, TP), and is the time required to trigger the timer.

Code block 2 Defining constants

```
(define-const SCAN_TIME Real 50)
(define-const K_DELAY Real 100)
```

Declaring Input Variables

Input variables are declared with the *declare-const* command, specifying a name and the type. Variables are declared with no definition, as these values will be assigned later in the program. In Code Block 3, we declared two variables of type Real: PV_OUT_t3, TSP_t3 and a variable named TRIP_LOGIC_t3 of type Bool.

Code block 3 Declaring input variables

```
(declare-const PV_OUT_t3 Real)
(declare-const TSP_t3 Real )
(declare-const TRIP_LOGIC_t3 Bool)
```

Defining FCs and FBs

FCs and FBs were defined with the command *define-fun* in SMT2. The syntax is (define-fun <name> () <type> <expression>), where <name> is the name of the function, <type> is the type of the value returned by the function, and <expression>, is the type of operation to be performed by the function. Expressions are built recursively from a set of basic forms. In the example given in Code Block 4, we defined two FCs (i.e. GE2_out_t3 and AND2_out_t3). The first function compares the values of the variables PV_OUT_t3 and TSP_t3 using the greater than or equal to operator. The result of the operation is a Boolean variable. The second function is a logical AND function, and also returns a Boolean variable. The expression first applies a logical not operation to the TRIP_LOGIC_t3 variable, and then performs a logical conjunction of the two Boolean variables.

Code block 4 Defining functions

```
(define-fun GE2_out_t3 () Bool (>= PV_OUT_t3 TSP_t3))
(define-fun AND2_out_t3 () Bool (and GE2_out_t3 (not TRIP_LOGIC_t3)))
```

Defining Function Conditions

Function conditions are a set of requirements that control the flow of data in the FBD program. Essentially these are the conditions that each of the FC and FB should meet to establish a path. Hence, conditions were defined as functions with the command *define-fun*, in a similar fashion to the definition of FCs and FBs. As an example, Code Block 5 defines two conditions as functions, namely C1_18_4 and C21_1_2. The set of conditions for C1_18_4 to meet are that AND2_out must be false, TON_et_t1 0, or AND2_out_t1 must be true, and TON_et_t1 greater than 0. The second condition C21_1_2 is met when TRIP_LOGIC is false and GE2_out is true.

Code block 5 Defining function conditions

```
(define-fun C1_18_4 () Bool (or (and (not AND2_out) (= TON_et_t1 0)) (and
AND2_out_t1 (> TON_et_t1 0))))
(define-fun C21_1_2 () Bool (or (not (not TRIP_LOGIC)) GE2_out))
```

Declaring DPCs

A DPC comprises a set of FC and FB conditions for a data path p. Structural coverage criteria are defined based on the DPC, as defined in Section 3.3.2. To this end, DPCs were declared as functions so that coverage-based requirements could be subsequently defined. They were declared following the syntax (declare-fun $< name >$ () <type>), where name indicates the name of the function, and type refers to the function return type. In this case, the variable names represent different branches of the data path, such as DPCp2_1_0, DPCp3_1_0, DPCp4_1_0, and DPCp4_2_0 (Code Block 6). The return variable was specified as a Boolean type, as they are assertion variables that will later be checked for satisfiability, returning either true or false.

Code block 6 Declaring data path conditions

```
(declare-fun DPCp2_1_0 () Bool)
(declare-fun DPCp3_1_0 () Bool)
(declare-fun DPCp4_1_0 () Bool)
(declare-fun DPCp4_2_0 () Bool)
```

Defining DPC Assertions

In this step, all coverage-based requirements were added to the DPC declared in the previous step. The solver then asserted that a DPC is a combination of different conditions (i.e. C1_18_0, C21_1_2). Assertions were defined using the *assert* command, in which a path is described as a combination of requirements using the logical AND operator. As an example, the DPCs or requirements for data paths (i.e. DPCp2_1_0, DPCp3_1_0, DPCp4_1_0, and DPCp4_2_0) were specified to be equal to the combined set of conditions (Code Block 7).

Code block 7 Defining assertions

```
(assert (= DPCp2_1_0 C22_18_3))
(assert (= DPCp3_1_0 (and C1_18_0 C21_1_2)))
(assert (= DPCp4_1_0 (and C1_18_0 C17_1_1)))
(assert (= DPCp4_2_0 (and C1_18_0 C17_1_1)))
```

Checking Assertions

The SMT2 solver then checked the satisfiability of the defined requirements in the data path. In Code Block 8 we call the *check-sat-assuming-model* function with the assumption that each of the specified data paths must be true (i.e. p2_1_0, p3_1_0, p4_1_0, and p4_2_0). In other words, this function checks if the logical formula represented by these variables and their definitions is satisfiable under the assumption that they are all true. If the function returns 'unsat', it means that the formula is not satisfiable for the specified set of assumptions.

Code block 8 Checking assertions

```
1: (check-sat-assuming-model (p2_1_0, p3_1_0, p4_1_0 p4_2_0) (true true true
   true))
```

Obtaining Input Data

If the result of the *check-sat-assuming-model* function is 'sat', it indicates that the constraints are satisfiable. In that case, the values of the input variables of the solution were retrieved with the function *get-value*, as indicated in Code Block 9. The returned solution contains a sequence of input values for each PLC iteration, which will be used to test the FBD for different coverage requirements. In Code Block 9, we retrieved values for 4 PLC scan cycles for the variables *PV_OUT* and *TSP*.

Code block 9 Obtaining input data

```
1: (get-value (PV_OUT))
2: (get-value (PV_OUT_t1))
3: (get-value (PV_OUT_t2))
4: (get-value (PV_OUT_t3))
5: (get-value (TSP))
6: (get-value (TSP_t1))
7: (get-value (TSP_t2))
8: (get-value (TSP_t3))
```

To sum up, the FBDTester 3.0 should generate a SMT2 file similar to the example file defined in Code Block 10. This file is then used by the SMT2 solver to solve coverage-based specified assertions. The SMT2 solver reads the file, defines constant data, declares input variables, and defines the specified FCs, FBs and function conditions for each of the scan cycles. The solver then declares DPCs as booleans and writes the assertions to be solved. The SMT2 solver attempts to solve these assertions, and if it finds a satisfiable solution, it returns the values of the input variables that led to the solution. These values represent the test case input data (as detailed in the results shown in Code Block 11).

Code block 10 Example SMT2 file

```
;Specifying the environment
(set-option :produce-unsat-model-interpolants true)
(set-option :produce-models true)
(set-logic QF_NIRA)
;Defining constants
(define-const SCAN_TIME Real 50)
(define-const K_DELAY Real 100)
(define-const TON_et_t2 Real 0)
;Declaring input variables 1st cycle
(declare-const PV_OUT_t1 Real)
(declare-const TRIP_LOGIC_t1 Bool )
(declare-const TSP_t1 Real )
;Defining FCs and FBs 1st cycle
(define-fun GE217_out_t1 () Bool (>= PV_OUT_t1 TSP_t1))
(define-fun AND21_out_t1 () Bool (and GE217_out_t1 (not TRIP_LOGIC_t1)))
(define-fun TON_et_t1 () Real (ite (and AND21_out_t1 false) (ite (< TON_et_t2 K_DELAY) (+ TON_et_t2 SCAN_TIME) K_DELAY) 0))
(define-fun TRIP_LOGIC_out_t1 () Bool (and false (>= TON_et_t1 K_DELAY)))
;Declaring input variables 2nd cycle
(declare-const PV_OUT Real)
(declare-const TSP Real)
;Defining FCs and FBs 2nd cycle
(define-fun TRIP_LOGIC () Bool TRIP_LOGIC_out_t1)
(define-fun GE217_out () Bool (>= PV_OUT TSP))
(define-fun AND21_out () Bool (and GE217_out (not TRIP_LOGIC)))
(define-fun TON_et () Real (ite (and AND21_out AND21_out_t1) (ite (< TON_et_t1 K_DELAY) (+ TON_et_t1 SCAN_TIME) K_DELAY) 0))
(define-fun TRIP_LOGIC_out () Bool (and AND21_out_t1 (>= TON_et K_DELAY)))
;Defining function conditions
(define-fun C1_18_0 () Bool (and (or (not AND21_out) (and AND21_out_t1 (>= TON_et_t1 K_DELAY))) (or (or AND21_out (and (not AND21_out_t1) (= TON_et_t1 0))) (and AND21_out_t1 (> TON_et_t1 0)))))
(define-fun C17_1_1 () Bool (or (not GE217_out) (not TRIP_LOGIC)))
(define-fun C21_1_2 () Bool (or (not (not TRIP_LOGIC)) GE217_out))
(define-fun C22_18_3 () Bool (or (and (not AND21_out) (= TON_et_t1 0)) (and AND21_out_t1 (> TON_et_t1 0))))
;Declaring DPCs
(declare-fun DPCp2_1_0 () Bool)
(declare-fun DPCp3_1_0 () Bool)
(declare-fun DPCp4_1_0 () Bool)
(declare-fun DPCp4_2_0 () Bool)
;Defining DPC assertions
(assert (= p2_1_0 C22_18_3))
(assert (= p3_1_0 (and C1_18_0 C21_1_2)))
(assert (= p4_1_0 (and C1_18_0 C17_1_1)))
(assert (= p4_2_0 (and C1_18_0 C17_1_1)))
;Checking assertions
(check-sat-assuming-model (p2_1_0, p3_1_0, p4_1_0 p4_2_0) (true true true true))
;Obtaining input data values
(get-value (PV_OUT))
(get-value (PV_OUT_t1))
(get-value (TSP))
(get-value (TSP_t1))
```

Code block 11 SMT2 file result

```
(input;PV_OUT 0)
(input;PV_OUT_t1 0)
(input;TSP 1)
(input;TSP_t1 0)
```

6.3.3 Yices 2 SMT2 Integration

This section outlines the changes made to the FBDTester 2.0 and MuFBDTester tools to integrate the new Yices 2 SMT2 solver. Each specific step is detailed, together with the challenges encountered during the integration process.

Unsatisfied Requirements

Yices 1 returns a solution even if some assumptions are not satisfiable. This is because it is used to return a solution for a satisfiable set of assertions, indicating the assumptions that were not satisfied. Yices 2 does not provide such functionality natively (whether Yices or SMT2 language are used or it is run from the API). Instead, Yices 2 is able to retrieve unsatisfied model interpolants, (i.e. *get-unsat-model-interpolant*). This function returns an unsat core, which is a subset of unsatisfiable assumptions that are already unsatisfiable. However, the returned solution does not necessarily provide all unsatisfied assertions. Therefore, the process of obtaining unsatisfied assertions in Yices 2 required modification to ensure the functionality of Yices 1, as indicated in Code Block 12.

Our algorithm first obtains the unsatisfied data paths (defined as *DPaths*), of which all will initially be unsatisfied. Next, it calls the *calculateSatisfied* function to determine the maximally satisfiable set of assertions, and retrieves both the satisfied and unsatisfied assertions. If there are any satisfied assertions, the *yicesExecuter* function is called to solve the problem and obtain the input values that led to the satisfiable solution. The outcome is a testcase, which will be added to the testSuite.

The algorithm then checks if there are any unsatisfied assertions left. If the number of unsatisfied assertions is zero or remains unchanged from one iteration to the next, the maximum satisfiable set of solutions is obtained and the algorithm finalises the calculation. In the opposite case, the solver continues to calculate and check for additional solutions to the unsatisfied assertions. The variable *previousUnsatSizeID* is used to keep track of the number of unsatisfied assertions from the previous iteration, whereas the variable *iter* indicates the number of iterations in the loop.

The algorithm defined in Code block 13 maximally satisfies the unsatisfied test assertions. Since Yices 2 does not natively support maxsat, assertions were incremen-

Code block 12 Calculating unsatisfied assertions

```
iter=0;
while true do
    unsatDpaths ← getUnsatDPaths
    satAssertIDs ← CalculateSatisfied(unsatDpaths)
    if satAssertIDs>0 then
        testCase(iter) ← yicesExecuter(satisfiedAssertions,'smt2')
        testSuite.add(testcase(iter))
    end
    if unsatAssertIDs = 0 then
        break
    end
    if unsatAssertIDs.size = previousUnsatSizeID then
        break
    else
        previousUnsatSizeID ← unsatAssertIDs.size
        iter++
    end
end
```

tally added while maintaining a set of satisfiable assertions, X. The algorithm starts by initialising X with the unsatisfied DPaths. Then, for each assertion A in the set of unsatisfied DPaths, the algorithm checks if the union of X and A is satisfiable by calling the yicesExecuter function. This process is repeated until all remaining unsatisfied data paths are maximally satisfied. The final result is the maximum satisfiable set of assertions, X.

Code block 13 Maximally satisfied assertions

```
function CALCULATESATISFIED(unsatDpaths, filename)
X ← unsatDpaths
for every assertion A in unsatDpaths do
    if yicesExecuter(X ∪ A, filename) = 'sat' then
            X ← X ∪ A
    end
end
return X
```

Yices 2 SMT2 Execution

To deploy the SMT2 solver in yicesExecuter, the executable of the solver had to be run with the generated SMT2 file in the command list. The command *–mcsat* was also included to derive solutions for non-linear arithmetic formulas. The command was executed as outlined in Code Block 14:

Code block 14 Yices 2 SMT2 execution command

```
executeCommand('./yices-smt2.exe'+filename+'–mcsat', '.');
```

Division Operators

In addition to including the command *–mcsat* in the commandList, other changes specific to SMT2 were required when using division operators. For example, a rational constant variable of type Real in the input was defined as a fraction using the syntax *Real (/ <numerator> <denominator>)*. Similarly, when the solution of a non-linear arithmetic operation is a rational number, the SMT2 solver will return the value in the same form.

Code block 15 Parsing return value

```
if s.contains('/') then
    String ratio[] = s.split('/');
    retvalue = Double.parseDouble(ratio[0]) / Double.parseDouble(ratio[1]);
else
    retvalue = Double.parseDouble(s);
end
```

Moreover, the way the value is parsed in the LogicStatement Java file of the FBDTester 2.0 and MuFBDTester was updated. The value was parsed as a fraction, as stated in Code Block 15.

Logic Statement Modifications

Finally, some further modifications were made to the operators and statements defined in the LogicStament Java file. As shown in Table 6.3, the <not equal> operator and the <if> statement have different syntax in Yices 1 and Yices 2 SMT2.

Table 6.3: Modified statements syntax

Solver + language	**<Not Equal> operator**	**<If> statement**
Yices 1 + Yices	/=	if
Yices 2 + SMT2	distinct	ite

6.3.4 Test Case and Mutant Generation

The new versions of FBDTester 2.0 and MuFBDTester (version 3.0) were enhanced by incorporating the new solver Yices 2 SMT2, as described in the previous sections. The new solver supports non-linear arithmetic operations and non-constant divisions, providing a more thorough testing methodology. Thus, we were able to generate coverage-based, mutation-based, and random-based test cases for a wide range of FBD programs, including those with non-linear functions. Test cases and mutants were generated as indicated in Steps 1-2 of the overall process presented in Section 6.2, which is further detailed in Figure 6.3.

On the one hand, we generated test cases as per the coverage criteria defined by Jee et al. [JYCB09] and later implemented by [SJB18] in the FBDTester 2.0. The tester first defines coverage-based requirements in the form of assertions in the SMT2 file. The Yices 2 SMT2 solver then generates test data that maximises the satisfaction of the specified test requirements for each PLC scan cycle. The outcome was three sets of coverage-based test cases, one for each criterion (i.e. BC, ICC, and CCC).

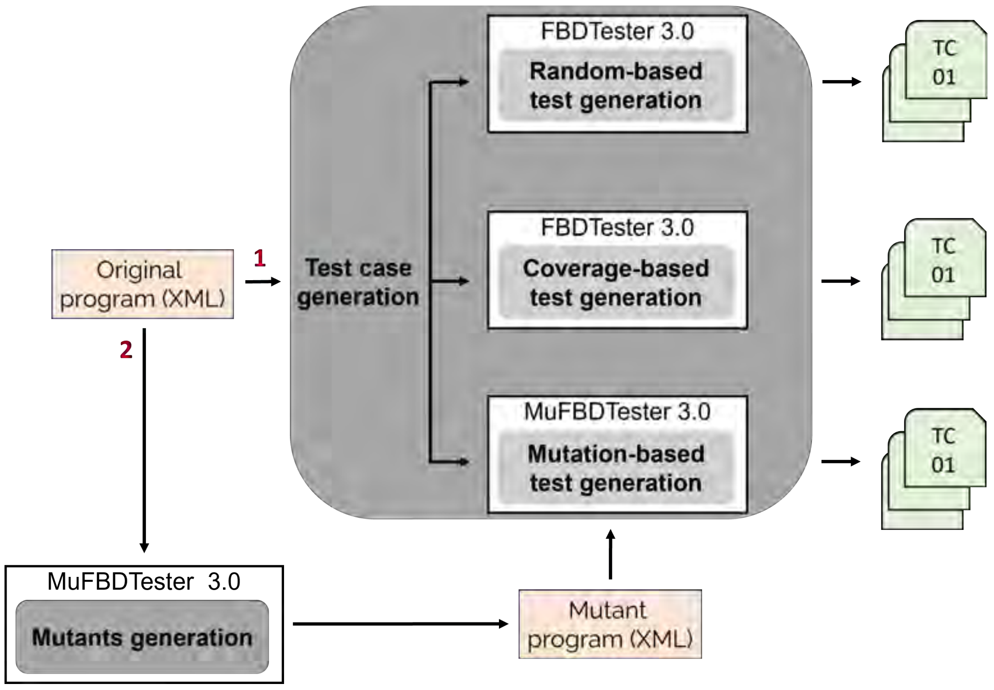


Figure 6.3: Test case and mutant generation

- Basic Coverage: every Dpath in the FBD program under test was tested at least once. Figure 6.4 shows a data path of an arithmetic FBD program.

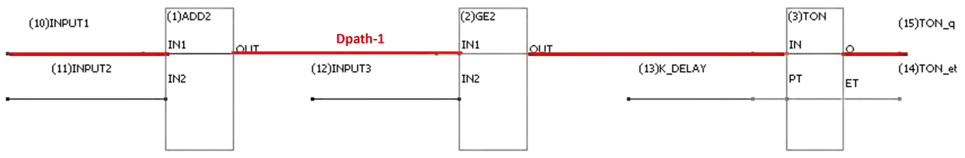


Figure 6.4: BC coverage example of an arithmetic FBD

■ Input Condition Coverage: Besides executing all the Dpaths as in Basic coverage, all the variations in values of the Boolean input edges were also exercised, as illustrated in Figure 6.5.

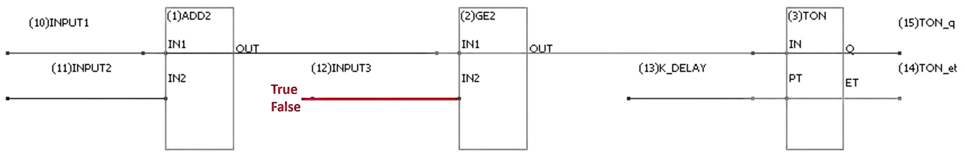


Figure 6.5: ICC coverage example of an arithmetic FBD

■ Complex Condition Coverage: Besides executing all the Dpaths as in Basic coverage, every Boolean edge (i.e input, internal and output edge) variation was also exercised, as depicted in Figure 6.6.

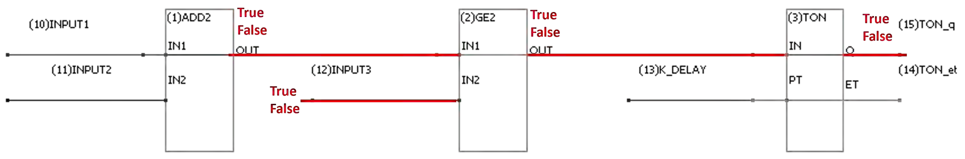


Figure 6.6: CCC coverage example of an arithmetic FBD

On the other hand, we generated mutation-based test cases and mutants, using the set of mutants defined in the MuFBDTester. Our MuFBDTester 3.0 tool generates test data specifically to detect and kill mutants, as per the original MuFBDTester. However, the MuFBDTester could also generate non-linear arithmetic test cases. The target program and the mutants were input into the Yices 2 SMT2 solver, which then determined if there was input data that could differentiate the output values of the mutant from those of the target program. As depicted in Figures 6.7-6.9, we generated 3 arithmetic mutants (i.e. multiplier, division, modulo) to replace the addition arithmetic function from the previous examples, with non-linear arithmetic operators.

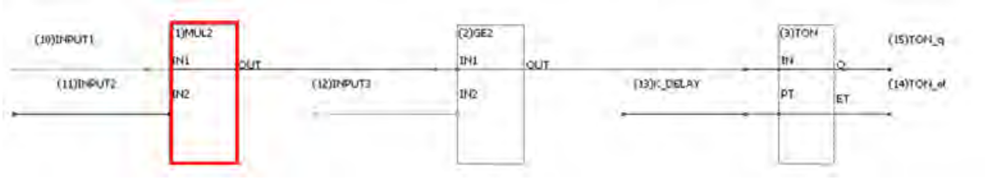


Figure 6.7: Addition operator replacement by a multiplier operator

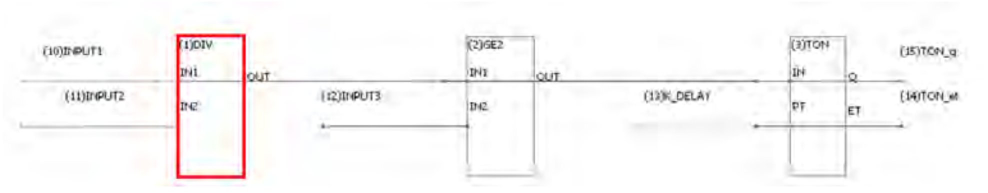


Figure 6.8: Addition operator replacement by a division operator

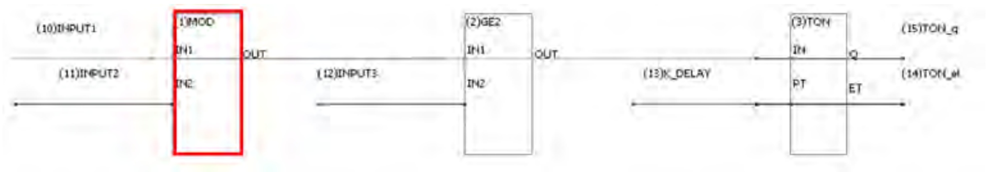


Figure 6.9: Addition operator replacement by a modulo operator

Lastly, a new test generation method was implemented in the FBDTester 3.0 to produce random test cases. Random tests were created with a range of scan cycles, as coverage was expected to increase with a higher number of iterations [SJB18]. The pseudocode for this process is described in Code Block 16, and further details on the *generateRandomTestSuite(usecase, ncycles, testDoc)* function are shown in Appendix C. The algorithm starts by generating a random number of cycles, i.e. ncycles, ranging from the second iteration to the maximum specified number of iterations. Next, the *generateRandomTestSuite* function is called to create the random test cases using the specified number of scan cycles. This function also obtains as inputs the use case (which indicates the FBD program name) and the file where input variables and constants for the FBD are defined (i.e. testDoc). The latter is required to determine the type of input variables to be generated (i.e. Boolean, Real).

It is important to note that a test case might require multiple cycles based on the characteristics of the FBD to fully test FBs. The minimum number of iterations were

Code block 16 Random test generation

```
for i = 0 to n do
    ncycles=random.nextInt(maxcycles)+1;
    generateRandomTestSuite(usecase, ncycles, testDoc)
end
```

derived from Song et al. [SJB18], which are indicated in Table 6.4. As a result, the minimum number of iterations required for an FBD program is determined by the highest minimum iteration number of all the FBs.

Table 6.4: Minimum iteration number of FBs

IEC 61131-3 FB	**Minimum number of iterations**
Timers	Pulse Time delay (PT) / scan_time + 1
Bistable elements	2
Edge detection	2
Counters	Processing Value (PV)

The solver result is a sequence of satisfiable input data that contains values for multiple scan cycles. For instance, a timer may require at least a minimum number of *PT/scan_time + 1* cycles to activate the output. Hence, if *PT = 100ms* and *scan_time = 50ms*, the solver would be limited to track 3 cycles back. The output of a test sequence would be described as indicated in Table 6.5:

Table 6.5: Example test sequence

_t3	0	true	-1
_t2	1	false	0
_t1	1	false	0
_t0	0	true	1

The iteration number is represented with the suffix *_t*, and indicates the number of cycles that have already occurred. For example *_t0* is the present cycle, *_t1* the previous cycle, and *_t2* two cycles back. This helps monitor the inputs over multiple cycles. The detailed solution states that the first set of inputs should be (0 true 1) in the first scan cycle *_t3*, (1 false 0) in the second *_t2*, (1 false 0) in the third *_t1*, and (0 true -1) for the last cycle *_t0*.

6.4 Cost-effectiveness Measures and Oracles Calculation

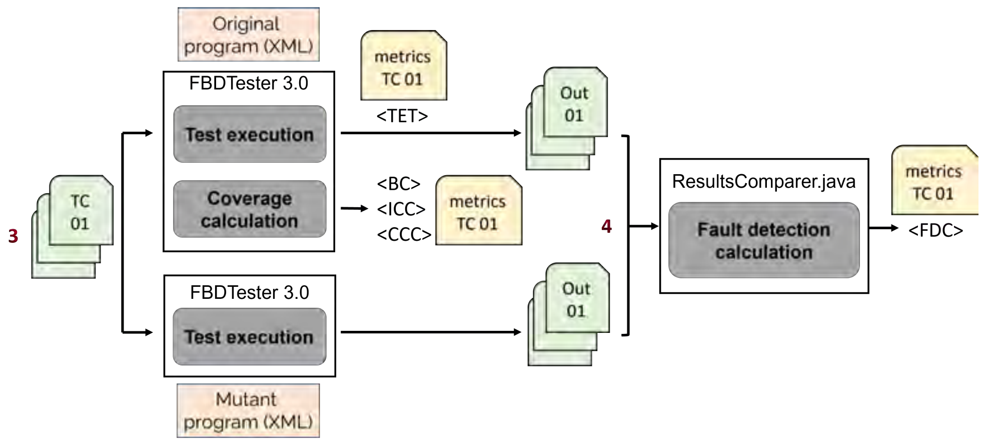


Figure 6.10: Test execution and metrics calculation approach

This section describes the approach for determining cost-effectiveness measures and oracles for the generated test cases, as defined in Steps 3-4 of the overall process presented in Section 6.2. Cost-effectiveness measures are used in Chapter 7 to optimise the efficiency and effectiveness of the test suite in detecting implementation errors. Moreover, expected outputs are generated to use as oracles when performing the SiL testing in Chapter 8.

We enhanced the FBDTester 2.0 [SJB18] to include test execution and the calculation of cost-effective metrics, as shown in Figure 6.10. This comprises coverage calculation (i.e. BC, ICC, CCC) of asserted data paths, fault detection capability based on mutant detection, and calculation of test case execution time.

6.4.1 Coverage Calculation

Structural coverage metrics can help reveal potential implementation errors in the code, as the amount of code which has been executed with a test case is measured. To this end, we developed a coverage calculation function, which calculates the BC, ICC, and CCC coverage for a given test sequence.

Coverage was calculated by checking if the requirements were met at every iteration, as different input values may lead to different coverage results per iteration. To support this approach, we generated a new Yices 2 SMT2 file with input data previously retrieved from the test generation process. The test case contains input values defined as constants, with the suffix *_t* indicating the iteration number. Similarly, when creating the SMT2 Yices file, these inputs were read from the test case and defined as constants. In this way, the input variables remained fixed throughout the entire execution of the FBD. An example test case and its corresponding translation to the SMT2 file are outlined in (Code Blocks 17-18).

Code block 17 Example test case

```
### scan cycle
50

### constants
K_DELAY, TSP_t3, TRIP_LOGIC_t3, PV_OUT_t3, TSP_t2, PV_OUT_t2, TSP_t1,
PV_OUT_t1, TSP, PV_OUT

### cTypes
100   0   false   0   0   0   0   0   1   0

### outputs
TRIP_LOGIC, TON_et

### inputs
TSP, TRIP_LOGIC, PV_OUT
```

Code block 18 Defining input data from the test case

```
  (define-const SCAN_TIME Real 50)

  (define-const K_DELAY Real 100)
  (define-const TSP_t3 Real 0)
  (define-const TRIP_LOGIC_t3 Bool false)
  (define-const PV_OUT_t3 Real 0)
  (define-const TSP_t2 Real 0)
  (define-const PV_OUT_t2 Real 0)
  (define-const TSP_t1 Real 0)
  (define-const PV_OUT_t1 Real 0)
  (define-const TSP Real 1)
  (define-const PV_OUT Real 0) =0
```

The SMT2 file not only contains constant definitions, but also FCs, FBs, DPCs, and definitions for data path variables, as described in Section 6.3.2. Coverage-based requirements were established, which are combinations of DPCs that comprise a data path. These requirements are specific to the coverage criteria (i.e. BC, ICC, CCC), with CCC the most complex criterion. As different inputs may activate various data paths, data path requirements could be met at any scan cycle. We therefore modified these requirements to become a combination of cyclic requirements. For instance, given that the DPCp1_2_0 requirement is the conjunction of C1, C2, and C3, the coverage requirement will be met if the presented conjunction is met at any PLC iteration, as denoted in Equation 6.1.

$$\begin{aligned} DPCp1\_2\_0 \;=\; & (C1 \;\&\; C2 \;\&\; C3) && : cycle\ t0 \\ & OR\ (C1\_t1 \;\&\; C2\_t1 \;\&\; C3\_t1) && : cycle\ t1 \\ & OR\ (C1\_t2 \;\&\; C2\_t2 \;\&\; C3\_t2) && : cycle\ t2 \\ & OR\ (C1\_t3 \;\&\; C2\_t3 \;\&\; C3\_t3) && : cycle\ t3 \end{aligned} \tag{6.1}$$

Code Block 19 was used to calculate the number of satisfied requirements (previously described in Section 6.3.3). It should be noted that the data path requirements are different for each criterion, in which DPaths include BC requirements, ICC_DPaths include BC and ICC requirements, and CCC_Dpaths include additional CCC requirements.

Code block 19 Satisfied requirements calculation

```
  satAssertIDs_BC ← calculateSatisfied(DPaths, SMT2file)
  satAssertIDs_ICC ← calculateSatisfied(ICC_DPaths, SMT2file)
  satAssertIDs_CCC ← calculateSatisfied(CCC_DPaths, SMT2file)
```

The coverage was then calculated by dividing the number of satisfied assertions by the total number of specified requirements, with the main difference being the specific data path requirements for each criterion, as indicated in the Equations 6.2-6.4 below:

$$BC\ coverage \;=\; \frac{\sum satAssertIDs\_BC}{\sum DPaths} \tag{6.2}$$

$$ICC\ coverage \;=\; \frac{\sum satAssertIDs\_ICC}{\sum ICC\_DPaths} \tag{6.3}$$

$$CCC\ coverage \;=\; \frac{\sum satAssertIDs\_CCC}{\sum CCC\_DPaths} \tag{6.4}$$

6.4.2 Execution Time Calculation

Testing PLCs is a time-consuming task that requires considerable effort. It involves interaction with the real hardware in the final stages, which can cause severe delays in deploying the system. Therefore, time plays a critical role when testing PLCs, as there is always a need to reduce commissioning time and shorten the time to market. For this reason, test case Execution Time (ET) was measured in this study. We define this calculation as the amount of time required by a test case to complete its execution of the program under test, as indicated in Equation 6.5.

$$ET_{TC} = t_{end} - t_{start} \tag{6.5}$$

in which,

$$\begin{aligned}
&t_{start} = System.currentTimeMillis();\\
&EvalTestSuite(testcase);\\
&t_{end} = System.currentTimeMillis();
\end{aligned}$$

The execution time in this case is the elapsed CPU time when running the test cases in the FBDTester 3.0 tool, and is measured in milliseconds.

6.4.3 Fault Detection Capability Calculation

To date, PLC testing has been carried out solely to validate the software against functional requirements, which does not take into account the internal design of the software itself. As this can lead to implementation errors, the main objective of the present study is to design tests that effectively identify implementation faults in the PLC programs. To this end, the Fault Detection Capability (FDC) of the test cases was obtained.

FDC was calculated by comparing the outputs of the original program with the mutant programs. A faulty program is detected if the test execution result differs from the source program, which is used as an oracle. In this case, a new SMT2 file for each of the mutant programs was generated.

The SMT2 file was created with the new mutant program as an input, as well as the test case in which the input values were generated. The solver then assessed the outputs based on FCs and FBs of the new program. To solve the output values, we defined the input data as constants in the SMT2 file, as well as the definition of the FCs and FBs of the program. There was no need to define DPCs and data path requirements, as these are only required for coverage calculation. However, to obtain the values of the output variables, we set the *get-value* function calls, as previously described in Code Block 9. The process was repeated for each of the test cases and mutant programs generated in Section 6.3.

To calculate the FDC of the tests, an in-house *ResultsComparer* Java script was developed. This script benchmarks the values of the output variables of the source program with the values retrieved from the mutants. If any of the variable values differ, the mutant is killed. The FDC is thus calculated as the ratio of the sum of killed mutants with respect to the total number of mutants (see Equation 6.6).

$$FDC = \frac{\sum killedMutants}{totalMutants} \tag{6.6}$$

6.5 Evaluation

This section evaluates the SMT2 solver proposed for generating test cases and their respective oracles for non-linear FBD programs. Four use cases are presented to verify that the enhanced test generation tools are able to generate test cases for a wide range of FBDs, including non-linear arithmetic functions. An extensive analysis of the generated cost-effective metrics to optimise the selection of the test cases is presented separately, in Chapter 7.

6.5.1 Case Studies

The first use case is a simple FBD program that was designed specifically to include a division operator, thereby named nonLinear. This FBD is composed of a division (DIV) non-linear arithmetic FC, a two inputs GE comparison FC, and a TON FB, as illustrated in Figure 6.11.

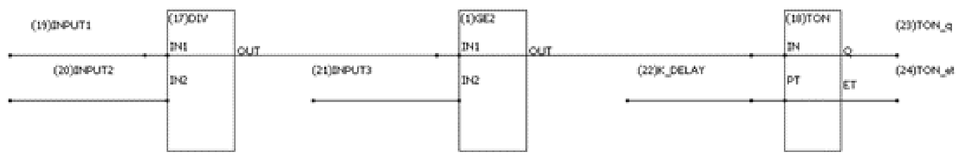


Figure 6.11: Simple non-linear arithmetic FBD program

The output variable TON_q is set to true when the output of GE2 has been true for a PT delay. The output variable TON_et is used to keep track of the elapsed time. As a result, if the division of the INPUT1 and INPUT2 is greater than or equal to INPUT3 for a PT time, the output TON_q will be true.

The remaining three use cases were selected to cover all IEC 61131-3 function groups supported by the Yices 1 solver, which have been widely used in the FBDTester 2.0 [SJB18] and the MuFBDTester [LJB22b].

- Use case 2 - LAUNCHER: FBD module for launching program codes.
- Use case 3 - simGRAVEL: FBD module used to control the amount of gravel delivered from a silo into a bin before being loaded onto a truck.
- Use case 4 - FFTD: Fixed-Falling Trip Decision FBD module of a bistable processor of the reactor protection system developed by KNICS [ind06].

In the tables below, we present some of the characteristics of the aforementioned use cases. Table 6.6 details the number of BC, ICC, and CCC coverage requirements of the use cases. Table 6.7 sets out the IEC 61131-3 function groups that are employed in each of the use cases.

Table 6.6: FBD characteristics of the use cases

Use case	#blocks	#inputs	#outputs	#BC	#ICC	#CCC
nonLinear	3	4	2	4	4	8
LAUNCHER	4	2	1	4	12	32
simGRAVEL	3	3	4	12	25	57
FFTD	29	12	8	126	185	774

Table 6.7: IEC 61131-3 groups employed in each of the use cases

IEC61131-3 group	nonLinear	LAUNCHER	simGRAVEL	FFTD
logic		X	X	X
timer	X		X	X
comparison	X			X
arithmetic				X
non-linear arithmetic	X			
selection				X
bistable		X		
edge trig		X		
counter			X	

Research Questions

The first objective was to validate the SMT2 solver for generating test cases for the IEC 63113-3 standard FCs and FBs supported by the baseline Yices 1 solver in the FBDTester 2.0 (RQ4), and in the MuFBDTester (RQ5). The second objective was to generate coverage-based test cases for non-linear arithmetic IEC 61131-3 functions (RQ6), which are not supported by Yices 1. Finally, the same objectives should be applicable when generating mutation-based test cases (RQ7). All these RQs were designed to address *Hypothesis 2*.

RQ4. Can the SMT2 solver in FBDTester 3.0 generate test cases for the IEC 61131-3 standard FCs and FBs that were supported by Yices 1 in FBDTester 2.0?
RQ5. Can the SMT2 solver in MuFBDTester 3.0 generate test cases for the IEC 61131-3 standard FCs and FBs that were supported by Yices 1 in MuFBDTester?
RQ6. Can the SMT2 solver in FBDTester 3.0 generate coverage-based test cases for non-linear arithmetic IEC 61131-3 functions?
RQ7. Can the SMT2 solver in MuFBDTester 3.0 generate mutation-based test cases for non-linear arithmetic IEC 61131-3 functions?

The research questions related to the effectiveness of the calculated cost-effective metrics are assessed in Chapter 7, whereas the practicality of generated test oracles is evaluated in Chapter 8.

Evaluation Setup

All the experiments were conducted on a Windows 10 Pro machine with 12 GB of RAM memory and Intel Core i5-7200U Processor. We used Eclipse IDE Java for Java Developers (version 4.21.0) to run the testing tools FBDTester 3.0 and MuFBDTester 3.0.

Table 6.8 details the use case employed in each of the research questions. RQ4 was evaluated with LAUNCHER, simGRAVEL, and FFTD, since this combination of use cases can test all IEC 61131-3 function groups described in Table 6.7. RQ5 was only tested with LAUNCHER, and simGRAVEL, as the FFTD comprises linear arithmetic operators (i.e. addition, subtraction) that could be replaced by non-linear arithmetic operators when generating mutants. This could not be benchmarked with the Yices 1 as it does not support such capability, and it was therefore addressed in RQ7. In addition, RQ6 and RQ7 were evaluated with the new non-linear FBD program, as it contains a non-constant division operator.

Table 6.8: Use cases employed in each of the research questions

Research Questions	nonLinear	LAUNCHER	simGRAVEL	FFTD
RQ4		X	X	X
RQ5		X	X	
RQ6	X			
RQ7	X			X

6.5.2 Results

RQ4. Can the SMT2 solver in FBDTester 3.0 generate test cases for the IEC 61131-3 standard FCs and FBs that were supported by Yices 1 in FBDTester 2.0?

To address RQ4, we assessed the test case generation capability of the FBDTester 3.0. The algorithm should generate test cases until maximum achievable coverage is achieved. The test size information, the accumulated total coverage, and elapsed time of the test cases are reported in Tables 6.9-6.11.

Table 6.9: Coverage-based test cases of LAUNCHER

Test case	**$\sum$ Coverage**	**Elapsed time**
BC_smt-based_evalTestSuite_001_1	0.50	1262 ms
BC_smt-based_evalTestSuite_001_2	1.00	585 ms
ICC_smt-based_evalTestSuite_001_1	0.33	2951 ms
ICC_smt-based_evalTestSuite_001_2	0.75	1781 ms
ICC_smt-based_evalTestSuite_001_3	0.83	843 ms
CCC_smt-based_evalTestSuite_001_1	0.28	6535 ms
CCC_smt-based_evalTestSuite_001_2	0.69	4665 ms
CCC_smt-based_evalTestSuite_001_3	0.78	1550 ms

Table 6.10: Coverage-based test cases of simGRAVEL

Test case	**$\sum$ Coverage**	**Elapsed time**
BC_smt-based_evalTestSuite_001_1	0.85	3584 ms
BC_smt-based_evalTestSuite_001_2	0.92	753 ms
ICC_smt-based_evalTestSuite_001_1	0.62	7839 ms
ICC_smt-based_evalTestSuite_001_2	0.72	3042 ms
CCC_smt-based_evalTestSuite_001_1	0.54	119317 ms
CCC_smt-based_evalTestSuite_001_2	0.62	11391 ms

Table 6.11: Coverage-based test cases of FFTD

Test case	$\sum$ **Coverage**	**Elapsed time**
BC_smt-based_evalTestSuite_001_1	0.65	18670 ms
BC_smt-based_evalTestSuite_001_2	0.75	6954 ms
BC_smt-based_evalTestSuite_001_3	0.87	5035 ms
BC_smt-based_evalTestSuite_001_4	0.94	2650 ms
BC_smt-based_evalTestSuite_001_5	1.00	1407 ms
ICC_smt-based_evalTestSuite_001_1	0.57	28564 ms
ICC_smt-based_evalTestSuite_001_2	0.70	12637 ms
ICC_smt-based_evalTestSuite_001_3	0.78	8942 ms
ICC_smt-based_evalTestSuite_001_4	0.83	6672 ms
ICC_smt-based_evalTestSuite_001_5	0.91	5211 ms
CCC_smt-based_evalTestSuite_001_1	0.41	145660 ms
CCC_smt-based_evalTestSuite_001_2	0.57	85865 ms
CCC_smt-based_evalTestSuite_001_3	0.63	63164 ms
CCC_smt-based_evalTestSuite_001_4	0.68	54468 ms
CCC_smt-based_evalTestSuite_001_5	0.76	47720 ms

The results show that the new solver was able to generate the test cases by incrementally increasing the number of test cases until maximum achievable coverage was reached. Time-wise, the elapsed time to generate the test cases was higher on the first test case than on subsequent test cases. This is because the first test case needed to maximally satisfy all coverage requirements, whereas the subsequent test cases only needed to address the remaining requirements.

To this end, the results demonstrate that the new SMT2 solver in FBDTester 3.0 can generate test cases for all the IEC 61131-3 function groups tested with Yices 1 solver in FBDTester 2.0.

RQ5. Can the SMT2 solver in MuFBDTester 3.0 generate test cases for the IEC 61131-3 standard FCs and FBs that were supported by Yices 1 in MuFBDTester?

To address RQ5, we benchmarked the generated test suite size of the MuFBDTester 3.0, with the baseline MuFBDTester, as the number of mutant operators, and hence mutation-based test cases should be the same. Test size information is set out in Table 6.12.

The number of mutation-based test cases was the same for both solvers. Hence, the results indicate that the new SMT2 solver can successfully generate mutation-

Table 6.12: Mutation test suite size of the use cases

Use case	Solver	#Mu test cases
LAUNCHER	**Yices 1**	16
	Yices 2 SMT2	16
simGRAVEL	**Yices 1**	13
	Yices 2 SMT2	13

based test cases for at least the same set of IEC 61131-3 function groups tested in the MuFBDTester.

RQ6. Can the SMT2 solver in FBDTester 3.0 generate coverage-based test cases for non-linear arithmetic IEC 61131-3 functions?

RQ6 evaluates the capacity of the new FBDTester 3.0 to test non-linear arithmetic functions. For this reason, a non-constant division operator was used in the subject program. As can be seen in Table 6.13, the new SMT2 solver could generate coverage-based test cases for the subject program, whereas Yices 1 could not derive any test cases.

Table 6.13: Coverage-based test cases of the non-linear FBD program

Test case	$\sum$ Coverage	Elapsed time
BC_smt-based_evalTestSuite_001_1	1.00	739 ms
ICC_smt-based_evalTestSuite_001_1	1.00	791 ms
CCC_smt-based_evalTestSuite_001_1	0.61	2725 ms
CCC_smt-based_evalTestSuite_001_2	1.00	1110 ms

RQ7. Can the SMT2 solver in MuFBDTester 3.0 generate mutation-based test cases for non-linear arithmetic IEC 61131-3 functions?

In RQ7 we evaluated the new MuFBDTester 3.0 with two arithmetic uses cases. The number of generated test cases of the new SMT2 solver was benchmarked against the number of tests generated with Yices 1, as indicated in Table 6.14.

Table 6.14: Mutation-based test suite size of arithmetic use cases

Use case	**Solver**	**#Mu test cases**
nonLinear	**Yices 1**	0
	Yices 2 SMT2	16
FFTD	**Yices 1**	121
	Yices 2 SMT2	133

The legacy MuFBDTester could not generate test cases for the non-linear FBD program, as Yices 1 does not support such capability. Instead, it could generate 121 test cases for the FFTD, which contains linear arithmetic functions. In contrast, the Yices 2 SMT2 solver could generate mutation-based test cases for both use cases. It is interesting to note that in the latter case, the number of generated tests was higher for the FFTD. This was due to non-linear Arithmetic Block Replacements (ABRs), as shown in Table 6.15.

Table 6.15: Non-linear arithmetic mutant operators generated with Yices 2 SMT2

Mutation operator	**Block ID**	**Mutation**
ABR	34	ADD -> MUL
ABR	34	ADD -> DIV
ABR	34	ADD -> MOD
ABR	35	ADD -> MUL
ABR	35	ADD -> DIV
ABR	35	ADD -> MOD
ABR	40	SUB -> MUL
ABR	40	SUB -> DIV
ABR	40	SUB -> MOD
ABR	48	SUB -> MUL
ABR	48	SUB -> DIV
ABR	48	SUB -> MOD

Non-linear arithmetic mutations were generated to replace the Addition FCs (i.e. block ID 34 and block ID 35) with division, multiplier, and modulo operators. Similarly, subtraction blocks (i.e. block ID 40 and block ID 48) were also replaced by the aforementioned non-linear arithmetic operators.

6.6 Conclusion

In this chapter, we focused on *Objective 2* of our methodology, which presents a test generation and evaluation approach to automatically generate test cases for most of the IEC 61131-3 function groups. We implemented a new solver (Yices 2 SMT2) in the test generation process to cover a wide range of IEC 61131-3 function groups, including non-linear arithmetic and non-constant division operators, which were not supported previously (*sub-contribution 1*). The test evaluation process consisted of cost-effective metrics calculation (*sub-contribution 2*) and test oracle generation (*sub-contribution 3*).

As a sanity check, we benchmarked the number of generated test cases of the new tools against the legacy tools for a broad set of IEC 61131-3 function groups. For evaluation, we employed use cases widely used in previous studies (i.e. LAUNCHER, simGRAVEL, FFTD). Specifically, the LAUNCHER use case consists of IEC 61131-3 logic, bistable, and edge trigger FCs. The simGRAVEL includes timer and counter FBs, and the FFTD features linear arithmetic, comparison, and selection FCs.

We evaluated the test generation capability of the new solver for non-linear arithmetic functions. This was analysed with two arithmetic use cases: a simple FBD that contains a division operator, and the FFTD previously used in the FBDTester 2.0 and MuFBDTester, as it includes addition and subtraction functions.

The results show that the new solver implemented in the FBDTester and MuFBDTester was able to generate test cases for at least the same IEC 61131-3 groups employed in previous studies. Furthermore, the solver supports the generation of non-linear arithmetic functions. As a result, this chapter addresses *Hypothesis 1* as:

Our findings demonstrate that coverage-based and mutation-based testing can be used to automatically generate test cases for most of the IEC 61131-3 function groups, including complex and non-linear arithmetic functions. This is attributed to the integration of the new Yices 2 SMT2 solver into the test generation process.

The generated cost-effective metrics are evaluated in Chapter 7 to assess if the calculated measures can cost-effectively select tests that maximise the detection of faults. Lastly, test outputs are used as oracles in Chapter 8 to benchmark the results in the industrial PLC simulation environment.

The following table summarises the contributions, objectives and hypotheses accomplished in this chapter. It is noteworthy that *Sub-Contributions 2-3* are preliminary tasks required to accomplish *Objectives 3-4*, respectively. These sub-contributions are therefore partially associated with *Objectives 3-4*, which are denoted with an asterisk (*) symbol.

Sub-contributions	**Contributions**	**Objectives**	**Hypotheses**
1	Technical Contribution 1	2	1
2	Technical Contribution 1	2, 3*	2*
3	Technical Contribution 1	2, 4*	3*

Part III

Test Optimisation

Chapter 7

Cost-effective Test Selection for FBD Programs

Contents

7.1 Introduction

A test selection approach using regression analysis is presented in this chapter. Our approach aims to cost-effectively test new FBD programs which undergo frequent changes to meet customer demand. To this end, a test selection problem was defined to meet the following requirements:

- Reduce execution time to minimise commissioning time.
- Maximise fault detection capability to prevent implementation errors.

This chapter focuses on our *Technical contribution 2* defined in Section 1.3, which aims at fulfilling the aforementioned requirements. The contribution is further divided into the following sub-contributions:

- **Sub-contribution 1:** A multi-objective search-based test case selection approach for FBD programs.
- **Sub-contribution 2:** Fitness functions objectives definition and formulation based on the cost-effective metrics calculated in Chapter 6.
- **Sub-contribution 3:** Empirical evaluation of the multi-objective test case selection approach at cost-effectively detecting faults.

The presented sub-contributions are directly linked to the contributions, objectives, and hypotheses described in Table 7.1:

Table 7.1: Chapter 7 contributions, objectives and hypotheses

Sub-contributions	Contributions	Objectives	Hypotheses
1, 2, 3	Technical Contribution 2	3	2

To this end, the main objective of this chapter is stated below:

Objective 3: Multi-objective search-based test selection approach to optimise the selection of tests in a cost-effective manner.

Section 7.2 details the overall approach. Section 7.3 describes our search-based multi-objective test selection approach which is designed to test recently modified or reconfigured FBD programs based on adequacy criteria (Sub-contributions 1-2). These latter were obtained from the cost-effective metrics presented in Chapter 6, which combine coverage criteria, test execution time, and fault detection capability. In total, 7 fitness functions were defined, using execution time as a cost metric.

We empirically evaluated fitness functions in 6 experimental scenarios, the details of which are outlined in Section 7.4 (Sub-contribution 3). The results are reported in Section 7.5. Section 7.6 details the threats to the validity of our experiment, and the conclusions are summarised in Section 7.7.

7.2 Overview

We propose a test selection approach for FBD testing, which is designed to meet a set of cost-effective objectives. The approach is based on search-based algorithms which are commonly employed to find the optimal solution to a problem. In our particular case, the search-based algorithm is required to reduce the time, whilst optimising the coverage and fault detection capability of the test suite (see Figure 7.1). These measures were retrieved from Chapter 6, which comprised FBD-specific coverage (i.e. BC, ICC, and CCC), mutation-based fault detection capability (see Section 7.3.1), and test case execution time. It should be noted that the tests were performed in Java, and hence differences in the execution time might be observed if run on a real PLC. Factors such as scan cycles, timing-related functions, clock jitter, and other internal processes would likely result in a proportionally greater execution time in a real hardware environment.

A multi-objective search-based test selection was applied to the test suite generated in Chapter 6. This comprises three different types of test cases: 1) coverage-based test cases that maximally satisfy the coverage-based test requirements of the program structure, 2) mutation-based test cases, and 3) random test cases. These test cases were enriched with the cost-effective metrics data obtained from Section 6.4. Cost-effective metrics were then combined to determine the fitness objectives in Section 7.3.2, which serve to guide the search algorithm. The mutant detection capability of the selected tests is analysed in Section 7.4.

7.3 Search-based Multi-Objective Test Selection

Multi-objective search-based objectives have been widely employed to optimise software engineering problems [RRV19], including test case selection. Multi-objective search-based test selection algorithms are guided by the feedback from more than one objective. To this end, we define three cost-effective metrics in Section 7.3.1, which are used to obtain the search objectives, namely fitness functions. In total, we derived 7 fitness functions (see Section 7.3.2), in which we considered two and three objectives in each of them. These were then integrated in the multi-objective

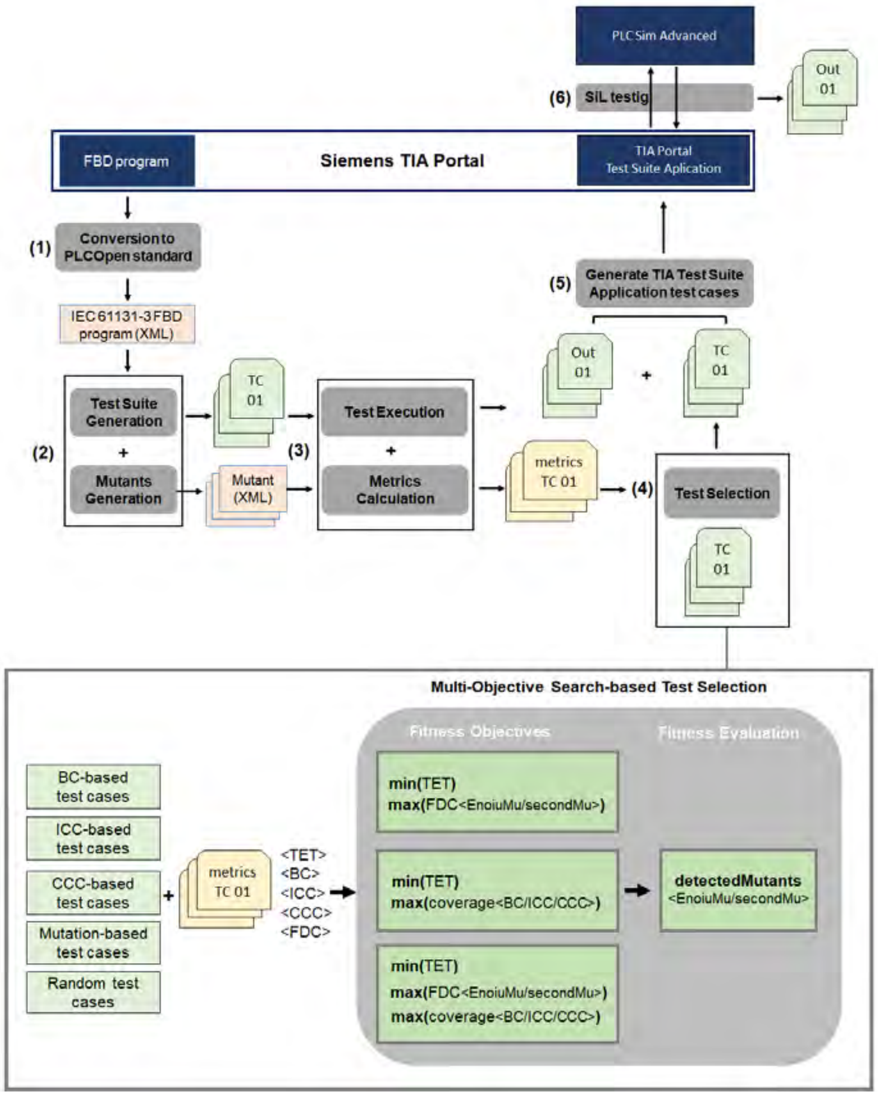

PLC Sim Advanced
(6) SiL testig
Out 01
FBD program
Siemens TIA Portal
TIA Portal Test Suite Aplication
(1) Conversion to PLCOpen standard
IEC 61131-3 FBD program (XML)
(5) Generate TIA Test Suite Application test cases
Out 01
TC 01
(2) Test Suite Generation + Mutants Generation
TC 01
Mutant (XML)
(3) Test Execution + Metrics Calculation
metrics TC 01
(4) Test Selection
TC 01
Multi-Objective Search-based Test Selection
Fitness Objectives
Fitness Evaluation
BC-based test cases
ICC-based test cases
CCC-based test cases
Mutation-based test cases
Random test cases
metrics TC 01
<TET> <BC> <ICC> <CCC> <FDC>
min(TET) max(FDC<EnoiuMu/secondMu>)
min(TET) max(coverage<BC/ICC/CCC>)
min(TET) max(FDC<EnoiuMu/secondMu>) max(coverage<BC/ICC/CCC>)
detectedMutants <EnoiuMu/secondMu>

Figure 7.1: Cost-effective test selection approach for FBD programs

search-based algorithm. Specifically, we used the NSGA-II [DPAM02], which is a commonly used algorithm in search optimisation problems.

7.3.1 Cost-effective Metrics

In this section, we formalise the cost-effective metrics of Chapter 6. These comprise three different coverage criteria (BC, ICC, and CCC), fault detection capability and test execution time, which have been formalised according to the notations defined below.

Formal Notations

Let $TS = \{tc_1, tc_2, ..., tc_N\}$ be a test suite (TS) comprising N test cases (tc) to test an FBD program. The quality and cost of the test suite is calculated with a set of f fitness functions (F) [PODPDL14] which must be satisfied when selecting test cases: $F = \{f_1, f_2, ..., f_p)$. The multi-objective test case selection algorithm is designed to select a subset of test cases from TS, such that $TS' = \{tc_1, tc_2, ..., tc_M\}$ is a subset of TS (i.e. $TS' \subseteq TS$). The NSGA-II returns a Pareto-optimal solution with respect to the fitness functions in F, with $M \leq N$.

We used a binary coding representation of the solution provided by the search algorithm, a method reported in several multi-objective test case selection studies [AWM$^+$19, YH07, PODPDL14, AVAS23, Arr22]. For instance, for a test suite of four test cases (i.e. $TS = \{tc_1, tc_2, tc_3, tc_4\}$), one possible solution s_k could be $s_k = \{1, 0, 1, 1\}$. This indicates that test cases tc_1, tc_3 and tc_4 are selected to be executed, and tc_2 is not.

Structural Coverage

Structural coverage metrics measure the amount of code executed with a test case, and as such are useful for identifying potential implementation errors in the code. We employed the data flow-based coverage criteria described by [SJB18] as effective fitness metrics, which include BC, ICC, and CCC. To determine the extent to which a test case covers all of the respective Dpath requirements, these criteria identify the executed Dpath requirement. If a data path requirement is satisfied, the specified coverage Test Requirement (TR) is asserted.

Multiple cycles based on the FBD characteristics may be necessary for the test case to completely test FBs. For example, a minimum number of *PT/scan_time* cycles are required by a Timer to activate the output. Hence, we determined coverage to be the combination of asserted test requirements in all scan cycles, as different input values could yield disparate coverage results.

Given that a solution s_k of the search algorithm denotes the Test Suite of selected test cases (i.e. $TS_{s_k} = \{tc_{s1}, tc_{s2}, ..., tc_{Ns}\}$, we defined the coverage criteria of the provided solution as $BC(s_k)$, $ICC(s_k)$, and $CCC(s_k)$, in a range of 0 to 1.

For BC (Equation 7.1), a value of 0 means that no Test Requirement (TR) containing Dpaths of the FBD was covered, while a value of 1 indicates that all Dpaths were covered. In the case of ICC (Equation 7.2), the coverage TR refers not only to the Dpaths, but also to boolean input edges (e_i). Lastly, CCC (Equation 7.3) is the most complex criterion, in which TR were defined based on the data paths, boolean input edges, boolean internal edges, and boolean output edges e_o.

$$BC_{s_k} = \frac{\sum_{i=1}^{N} TR_{s_i} < Dpath >}{TR < Dpath >} \tag{7.1}$$

$$ICC_{s_k} = \frac{\sum_{i=1}^{N} TR_{s_i} < Dpath, e_i >}{TR < Dpath, e_i >} \tag{7.2}$$

$$CCC_{s_k} = \frac{\sum_{i=1}^{N} TR_{s_i} < Dpath, e_i, e_o >}{TR < Dpath, e_i, e_o >} \tag{7.3}$$

The objective of our algorithm is to **maximise** these metrics.

Fault Detection Capability

Although a standard metric for determining the quality of a test suite is white-box coverage, a number of authors have observed that higher coverage may not always ensure higher fault detection [IH14, GSWH15]. For this reason, an alternative approach was proposed by [PWAY16], in which the quality of a test suite is calculated based on the FDC of each test case. In this scenario, a solution s_k is given by the search algorithm that denotes the Test Suite of selected test cases (i.e. $TS_{s_k} = \{tc_{s_1}, tc_{s_2}, ..., tc_{s_N}\}$, and the FDC is obtained with Equation 7.4.

$$FDC_{s_k} = \frac{\sum_{i=1}^{N} SuccRat_{tc_{s_i}}}{M} \tag{7.4}$$

where $SuccRat_{tc_{s_k}}$ is the success rate of the i-th test case in TS_{s_k}. The success rate can be calculated in accordance with the available information (e.g. previous information of failures [AVAS23], or FDC of each individual test case [PWAY16]). Since historical data was not available, we employed mutants in the present study. If the test execution results deviate from the source program, a mutant is killed. Thus, for a given set of mutants, the number of mutants a test case detects determines its success rate.

The objective of our algorithm is to **maximise** the FDC of a given test suite.

Total Execution Time

The use of Execution Time as a cost objective is widespread, and has been reported in several multi-objective test selection and optimisation studies [ZHL$^+$16, YH07, Har11, WSKR06]. Since time is critical in PLC testing, in the present study we employed the TET of the selected test suite as a cost measure.

If ET is the amount of time required by a test case to complete its execution of the program under test, we determined the TET of the selected test suite with Equation

7.5.

$$TET_s = \frac{\sum_{i=1}^{N} ET_{ts_i}}{ET_{ini}} \tag{7.5}$$

where ET_{ts_i} is the execution time of the selected test case, and ET_{ini} the ET of the initial TS.

The aim of our algorithm is to **minimise** the TET of a given test suite.

7.3.2 Fitness Functions

The scalability constraints of the search-based algorithms meant that the maximum number of cost-effective objectives was restricted to three. Combining these gave rise to seven fitness functions. We maintained TET as a cost metric in each fitness function, and combined this with the effectiveness metrics, as set out in Table 7.2.

Table 7.2: Fitness function objectives

fitness function	Objective 1	Objective 2	Objective 3
f1	TET	FDC	
f2	TET	BC	
f3	TET	ICC	
f4	TET	CCC	
f5	TET	FDC	BC
f6	TET	FDC	ICC
f7	TET	FDC	CCC

The employed search-based algorithm aims to minimise objectives. Thus, the fitness functions for BC, ICC, CCC, and FDC were inverted to maximise the ratios.

7.4 Evaluation

In this section, we describe the research questions to be addressed. The case studies selected to test the FBD programs, evaluation metrics, the algorithm setup, and statistical tests employed in the experiment are also detailed.

7.4.1 Research Questions

We defined three RQs. In comparison with random search, the first RQ asks if our problem to solve is complex enough to require search algorithms. The second RQ seeks to identify the fitness function which detects faults in the most cost-effective manner. The aim of the final RQ is to evaluate any improvement in performance, as compared to the initial test suite. These RQs are directly linked to our research

Hypoteshis 3, which claims that the search-based algorithm optimises the selection of test cases in a cost-effective manner.

RQ8. How does the multi-objective search-based algorithm perform as compared to RS?
RQ9. Which of the defined cost-effectiveness metrics performs best?
RQ10. To what extent can our approach reduce TET and what impact does it have on the fault detection rate?

7.4.2 Case Studies

The empirical evaluation consisted of three case studies, which have been widely used in previous studies [SJB16, SJB18, JSC^{+}14]:

- LAUNCHER: a module used for launching programs in a system.
- FFTD: a fixed-falling trip decision module of a reactor protection system.
- FRTD: a module that describes a fixed-rising trip decision of a reactor protection system.

FFTD and FRTD are parts of the PLC code of a bistable processor of the reactor protection system developed by KNICS [ind06]. These two case studies are quite large in terms of the number of blocks, inputs and outputs, and include 5 types of IEC 61131-3 function groups. The LAUNCHER was employed to cover IEC 61131-3 function groups (i.e. bistable, edge trig) which were not addressed in the other modules. Table 7.3 details the key characteristics of the selected case studies.

7.4.3 Evaluation Metrics

In our study, we utilised the revisited hypervolume (HV) quality indicator to assess RQ8 and RQ9, as reported in [AVAS23, AWM^{+}19]. The HV measures the volume of the objective space that is dominated by the solutions in a Pareto-frontier set [ABBZ09]. If the HV value is high, then the set of solutions encompasses a large area of the space, thereby yielding a better trade-off. Hence, the higher the HV value, the better the performance of the metrics. The revisited HV evaluates each solution returned by the Pareto-frontier to determine the ratio of detected faults to the total cost (i.e. TET). Using this information a new Pareto-frontier is obtained, which maximises the ratio of the detected faults and minimises execution time. The HV is then calculated with the new Pareto-frontier.

Table 7.3: Key characteristics of the selected case studies

		LAUNCHER	**FFTD**	**FRTD**
#blocks		4	29	29
#inputs		2	12	12
#outputs		1	8	8
IEC61131-3	**logic**	AND	AND, OR	AND, OR
	timer	-	TON	TON
	comparison	-	LE, LT, GT	GE, LT
	arithmetic	-	ADD, SUB	ADD, SUB
	selection	-	SEL	SEL
	bistable	SR	-	-
	edge trigger	R_TRIG	-	-
#BC requirements		4	126	126
#ICC requirements		12	186	186
#CCC requirements		32	902	902
#BC test cases		2	4	4
#ICC test cases		20	5	5
#CCC test cases		20	30	30
#mutant test cases		8	83	120
#random test cases		94	86	95
#Enoiu mutants		5	149	123
#second-order mutants		147	1000	1000
#max num of cycles		20	19	20

The evaluation process requires information of faults, however in our case studies real fault information was not available. For this reason, we employed mutation testing, which is reported in the literature as an appropriate substitute for real faults [JJI$^+$14]. Since mutants were also required to obtain the FDC of test cases, we used two sets of mutants (Enoiu mutants and second-order mutants) to calculate the FDC, which was then employed to measure the revisited HV:

- Enoiu mutants: new mutant operators proposed by Enoiu et al. [EČO$^+$16], which include Feedback Loop Insertion Operator (FIO), Logical Block Insertion Operator (LIO), Logical Block Deletion Operator (LDO), Value Replacement Operator-Improved (VRO-I), Logical Block Replacement Operator-Improved (LRO-I). In addition, the Variable Replacement Operator (VRO) was also implemented.

- Second-order mutants: mutants generated by combining two first-order mutants. The second-order set of mutants was derived from Jee et al.'s [JSB18] mutation operator set.

Hence, test cases with FDC based on Enoiu et al. [ESv$^+$16] mutants were evaluated with second-order mutants and vice versa.

The primary objective of test case selection is to minimise TET while maintaining the overall ability to detect faults. This describes the cost-benefit analysis set out in RQ10. For this purpose, we evaluated each of the Pareto-frontier solutions individually and checked their mutation score. From each of these, the highest-scoring solution was selected, and in the case of more than one solution, the one with the lowest TET was retrieved.

7.4.4 Algorithm Setup

We implemented our approach on the PlatEMO [TCZJ17] open-source platform, which is a widely-employed multi-objective search library for MATLAB. The implemented algorithm setup comprises:

- **Search-based algorithm:** NSGA-II, which has been widely used for multi-objective optimisation problems [DPAM02, VPS21].
- **Population size:** 100 (set as default value at PlatEMO [TCZJ17]).
- **Maximum generation:** 250 (as employed in several studies with a similar problem [YH07, AWM$^+$19, AVAS23]).
- **Number of objectives:** the cost-effective objectives were limited to three, as NSGA-II has been found to scale up to 3 objective functions [PKT17, LLTY15].

We employed a standard single point crossover with a rate of 0.8 as implemented by [AVAS23, AWM$^+$19]. The mutation of a variable was also obtained with a standard probability of 1/N, in which N is the total number of test cases.

7.4.5 Experimental Setup

A total of 6 experimental scenarios were assessed (3 case studies x 2 mutation detection evaluations). For each of the experimental scenarios, we measured the effectiveness of 7 fitness functions. Since randomised algorithms (i.e. search algorithms) return a different result each time the same problem instance is executed, each of the experimental scenarios was performed 50 times. This is in accordance with the guidelines of Arcuri and Briand [AB11].

The Vargha and Delaney $\hat{A}_{12}$ test was used to conduct the statistical analysis in each of the experiments, and the results were classified into various levels of significance difference, as per Romano et al. [RKC$^+$06]. We also employed the

Kruskal-Wallis test [MN10] to evaluate the statistical difference between the fitness functions.

We used Matlab 2020b to run the test selection algorithm on the experimental scenarios and perform statistical tests. In addition, we used Spyder IDE (5.1.5) for Python 3.9 to obtain the statistical figures. All these experiments were conducted on a Windows 10 Pro machine with 12 GB of RAM memory and Intel Core i5-7200U Processor.

7.5 Results and Discussion

This section reports the results of the experimental setup with the three case studies (LAUNCHER, FFTD, and FRTD). The following subsections are dedicated to the three research questions that were defined for each of the experimental scenarios described previously.

7.5.1 RQ8. How does the multi-objective search-based algorithm perform as compared to RS?

The goal of RQ8 is to demonstrate that our approach can effectively make a non-trivial selection of minimal-cost test cases. To this end, we compared the performance of the multi-objective search-based algorithm NSGA-II against the baseline algorithm RS. Specifically, the cost-effectiveness of 7 fitness functions was assessed by comparing NSGA-II and RS, as illustrated in Figure 7.2. The performance was measured with the revisited HV scores (ranging from 0 to 1). The HV of all fitness combinations for each of the experimental studies is also presented in the following boxplots. These results were supported by statistical analysis.

To evaluate the significance of the difference, the Vargha and Delaney $\hat{A}_{12}$ values were calculated and categorised by levels of effect-size. This scale was established by Romano et al. [RKC^{+}06] and is divided into: small, medium, and large. The calculations were obtained by benchmarking the results of both algorithms (A = NSGA-II, B = RS), which determined if the difference between A and B was significantly high (i.e. A++, B++), medium (i.e. A+, B+), or low (i.e. A, B). The values A, A+, A++ indicate that the results favour NSGA-II, in contrast B, B+, B++ indicate that RS outperformed NSGA-II. Cases in which both algorithms performed equally were classified as (=) value. Additionally, we computed the statistical difference indicator p-value (in parenthesis), and highlighted in bold those that were statistically significant (p-value≤ 0.05). These results are reported in Table 7.4 and Table 7.5.

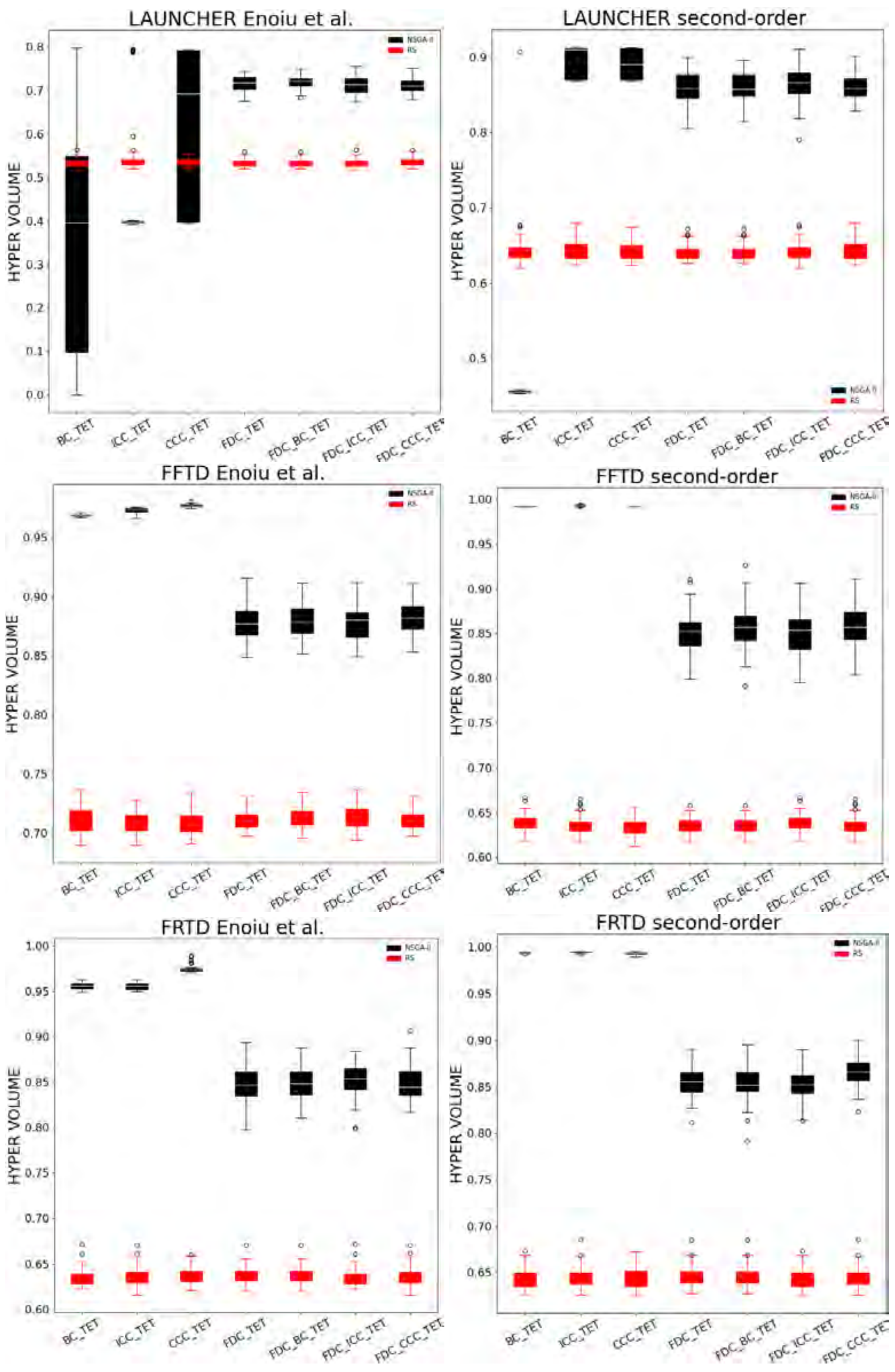


Figure 7.2: NSGA-II vs RS HV evaluation boxplots

According to the HV results, the multi-objective search-based NSGA-II algorithm consistently outperformed the RS algorithm in the vast majority of experimental scenarios. According to the $\hat{A}_{12}$ results, the performance of the NSGA-II algorithm was significantly better than RS (i.e. A++) in 40 out of 42 experiments (7 fitness functions x 6 experimental scenarios). Analysis of the LAUNCHER use case, however, yielded different results when using BC_TET fitness metric, as indicated in Tables 7.4-7.5. Here, the performance of the RS algorithm was slightly better (B) than NSGA-II with the Enoiu et al. mutants, and considerably better (i.e. B++) than NSGA-II with second-order mutants.

Table 7.4: p-values and Vargha and Delaney $\hat{A}_{12}$ results based on Romano et al. classification, where A=NSGA-II and B=RS. Evaluation with Enoiu et al. mutants

	LAUNCHER Enoiu et al.	**FFTD Enoiu et al.**	**FRTD Enoiu et al.**
	A++/A+/A/=/B/B+/B++	**A++/A+/A/=/B/B+/B++**	**A++/A+/A/=/B/B+/B++**
f1	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**
f2	**0/0/0/0/1/0/0 (0.05)**	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**
f3	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**
f4	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**
f5	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**
f6	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**
f7	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**

As a further analysis, the results of the setup that achieved the highest average HV are summarised in Table 7.6. The NSGA-II algorithm presented the best performance with *f1, f3, f4, f5, f6,* and *f7* fitness metrics in all of the case studies in relation to RS. This could indicate that NSGA-II is a good strategy when using these combinations at detecting faults in the FBD programs under study, which shows that it is not a trivial problem. On the contrary, the performance of the NSGA-II algorithm when using the *f2* function (i.e. BC_TET) was worse than RS. Hence, RS seems a more suitable strategy when using *f2* metric in the LAUNCHER case study.

In response to RQ8, the findings demonstrate that the performance of the multi-objective search-based NSGA-II algorithm is markedly better than the baseline RS in 95.24% of the experiments.

Table 7.5: p-values and Vargha and Delaney $\hat{A}_{12}$ results based on Romano et al. classification, where A=NSGA-II and B=RS. Evaluation with second-order mutants

	LAUNCHER Second-order	**FFTD Second-order**	**FRTD Second-order**
	A++/A+/A/=/B/B+/B++	**A++/A+/A/=/B/B+/B++**	**A++/A+/A/=/B/B+/B++**
f1	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**
f2	**0/0/0/0/0/0/1 (0.00)**	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**
f3	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**
f4	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**
f5	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**
f6	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**
f7	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**	**1/0/0/0/0/0/0 (0.00)**

Table 7.6: Summary of best algorithm results per fitness configuration

Case study	**Mutants**	**Algorithm**	**Fitness function**
LAUNCHER	**Enoiu et al.**	NSGA-II	f1, f3, f4, f5, f6, f7
		RS	f2
	Second-order	NSGA-II	f1, f3, f4, f5, f6, f7
		RS	f2
FFTD	**Enoiu et al.**	NSGA-II	f1, f2, f3, f4, f5, f6, f7
		RS	-
	Second-order	NSGA-II	f1, f2, f3, f4, f5, f6, f7
		RS	-
FRTD	**Enoiu et al.**	NSGA-II	f1, f2, f3, f4, f5, f6, f7
		RS	-
	Second-order	NSGA-II	f1, f2, f3, f4, f5, f6, f7
		RS	-

7.5.2 RQ9. Which of the defined cost-effectiveness metrics performs best?

The objective of RQ9 is to extract the optimal fitness function. For this purpose, we compared the performance of each of the fitness functions with the NSGA-II algorithm. Once again, the statistical analysis was conducted with the Vargha and Delaney $\hat{A}_{12}$ test, as set out in Tables 7.7-7.9. In this case, we employed the $\hat{A}_{12}$ test to compare the performance of the fitness functions calculated by the HV performance indicator.

An $\hat{A}_{12}$ value above 0.5 denotes which of the fitness function in the rows performs better than the fitness functions in the columns. When two fitness functions present a similar performance, the $\hat{A}_{12}$ value is 0.5. Any statistical difference between the fitness functions was determined by the Kruskal-Wallis test [MN10]. A statistically significant difference presents a p-value less than or equal to 0.05.

Table 7.7: Vargha and Delaney $\hat{A}_{12}$ values and p-values for the fitness functions – NSGAII LAUNCHER

		f1	**f2**	**f3**	**f4**	**f5**	**f6**	**f7**
Enoiu et al.	**f1**	-	**0.82 (0.00)**	0.30 (0.47)	0.34 (0.07)	0.46 (0.92)	0.55 (0.89)	0.61 (0.77)
	f2	**0.18 (0.00)**	-	**0.22 (0.00)**	**0.31 (0.00)**	**0.18 (0.00)**	**0.18 (0.00)**	**0.18 (0.00)**
	f3	0.70 (0.47)	**0.78 (0.00)**	-	**0.65 (0.01)**	0.70 (0.53)	0.70 (0.39)	0.70 (0.31)
	f4	0.66 (0.07)	**0.69 (0.00)**	**0.35 (0.01)**	-	0.66 (0.06)	0.66 (0.10)	0.66 (0.13)
	f5	0.54 (0.92)	**0.82 (0.00)**	0.30 (0.53)	0.34 (0.06)	-	0.59 (0.81)	0.66 (0.69)
	f6	0.45 (0.89)	**0.82 (0.00)**	0.30 (0.39)	0.34 (0.10)	0.41 (0.81)	-	0.55 (0.88)
	f7	0.39 (0.77)	**0.82 (0.00)**	0.30 (0.31)	0.34 (0.13)	0.34 (0.69)	0.45 (0.88)	-
Second order	**f1**	-	**0.98 (0.00)**	**0.15 (0.00)**	**0.18 (0.00)**	0.48 (0.78)	0.44 (0.44)	0.47 (0.67)
	f2	**0.02 (0.00)**	-	**0.01 (0.00)**	**0.01 (0.00)**	**0.02 (0.00)**	**0.02 (0.00)**	**0.02 (0.00)**
	f3	**0.85 (0.00)**	**0.99 (0.00)**	-	0.51 (0.61)	**0.85 (0.00)**	**0.81 (0.00)**	**0.87 (0.00)**
	f4	**0.82 (0.00)**	**0.99 (0.00)**	0.49 (0.61)	-	**0.83 (0.00)**	**0.77 (0.00)**	**0.85 (0.00)**
	f5	0.52 (0.78)	**0.98 (0.00)**	**0.15 (0.00)**	**0.17 (0.00)**	-	0.44 (0.62)	0.49 (0.88)
	f6	0.56 (0.44)	**0.98 (0.00)**	**0.19 (0.00)**	**0.23 (0.00)**	0.56 (0.62)	-	0.56 (0.73)
	f7	0.53 (0.67)	**0.98 (0.00)**	**0.13 (0.00)**	**0.15 (0.00)**	0.51 (0.88)	0.44 (0.73)	-

The A_{12} values in Tables 7.7-7.9 show that f4 – the combination of CCC and TET – is the best performing cost-effective metric in 5 out of 6 use cases. Similarly, analysis of the combination of the three objectives *f5, f6*, and *f7* reveals that the performance of the latter (i.e. the combination of CCC, FDC, and TET) was slightly better than *f5 and f6*. The ICC criterion (i.e. *f3*) presented similar performance to CCC when second-order mutants were used. BC criterion also performed well in FFTD and FRTD with second-order mutants. However, it reported the worst performance in the case of the LAUNCHER. Finally, the FDC related metrics (i.e. *f1, f5, f6, f7*), were found to perform worse than all coverage-based metrics in the case of the FRTD and FFTD. The LAUNCHER case, however, saw better results with these metrics than with *f1* (i.e. the combination of BC and TET).

Our findings show that utilising the combination of CCC and TET, the algorithm delivers cost-effective performance in all case studies. As a result, we can conclude that CCC is an effective method of selecting test cases. No measurable improvement in terms of performance was observed when using three objectives, i.e. the combination of TET, FDC, and data flow coverage-based metrics (BC, ICC, CCC). As such, the fault detection capability does not appear to present any significant improvement in performance. Calculating FDC is a time-consuming task, particularly when using a large set of mutants like second-order mutants. Thus, in the interest of cost-cutting, we recommend using a combination of CCC and TET, instead of the three objective fitness functions.

7.5.3 RQ10. To what extent can our approach reduce TET and what impact does it have on the fault detection rate?

RQ10 is designed to evaluate the cost-benefit of the multi-objective search-based algorithm. Accordingly, the results of the last population with the highest mutation score and lowest execution time were collected for each of the experimental scenarios. We used the last population set to retrieve the test suite of the multi-objective algorithm, since it delivers the highest HV scores. The average values of these results were then compared to the initial test suite. Table 7.10 sets out the reduced TET, number of test cases, and the ratio of detected mutants of the multi-objective search-based algorithm against the initial test suite. It is evident from these results that the FDC-related metrics (i.e. FDC_TET_BC, FDC_TET, FDC_TET_ICC, FDC_TET_CCC) obtained the maximum achievable mutation score with the given test suite (i.e. 100% for FFTD Enoiu et al, 99.80% for FFTD second-order, 100% for FRTD Enoiu et al, 99.80% for FRTD second-order, 80% for LAUNCHER Enoiu et al, 95.83% for LAUNCHER second-order). In all cases, the execution time was substantially reduced by up to 70.80%, 84.14%, 77.10%, 84.11%, 89.87%, and 89.78%, respectively.

In contrast, the BC_TET and ICC_TET metrics led to decreased mutation scores. BC_TET scored 78.50%, 78.47%, 80.77%, 98.87%, 39.20%, and 45.83%, respectively. ICC_TET reported a marginally better performance, with a mutation score of 83.30%, 83.30%, 98.82%, 99.04%, 40%, and 89.92%.

Table 7.8: Vargha and Delaney $\hat{A}_{12}$ values and p-values for the fitness functions – NSGAII FFTD

		f1	**f2**	**f3**	**f4**	**f5**	**f6**	**f7**
Enoiu et al.	**f1**	-	**1.00 (0.00)**	**0.82 (0.00)**	**0.04 (0.00)**	0.48 (0.66)	0.41 (0.19)	0.50 (0.89)
	f2	**0.00 (0.00)**	-	**0.19 (0.00)**	**0.00 (0.00)**	**0.00 (0.00)**	**0.00 (0.00)**	**0.00 (0.00)**
	f3	**0.18 (0.00)**	**0.81 (0.00)**	-	**0.00 (0.00)**	**0.15 (0.00)**	**0.11 (0.00)**	**0.21 (0.00)**
	f4	**0.96 (0.00)**	**1.00 (0.00)**	**1.00 (0.00)**	-	**0.96 (0.00)**	**0.93 (0.00)**	**0.95 (0.00)**
	f5	0.52 (0.66)	**1.00 (0.00)**	**0.85 (0.00)**	**0.04 (0.00)**	-	0.42 (0.38)	0.54 (0.76)
	f6	0.59 (0.19)	**1.00 (0.00)**	**0.89 (0.00)**	**0.07 (0.00)**	0.58 (0.38)	-	0.60 (0.23)
	f7	0.50 (0.89)	**1.00 (0.00)**	**0.79 (0.00)**	**0.05 (0.00)**	0.46 (0.76)	0.40 (0.23)	-
Second order	**f1**	-	**0.00 (0.00)**	**0.00 (0.00)**	**0.00 (0.00)**	**0.40 (0.03)**	0.50 (0.85)	**0.40 (0.03)**
	f2	**1.00 (0.00)**	-	0.50 (0.67)	0.68 (0.90)	**1.00 (0.00)**	**1.00 (0.00)**	**1.00 (0.00)**
	f3	**1.00 (0.00)**	0.50 (0.67)	-	0.67 (0.77)	**1.00 (0.00)**	**1.00 (0.00)**	**1.00 (0.00)**
	f4	**1.00 (0.00)**	0.32 (0.90)	0.33 (0.77)	-	**1.00 (0.00)**	**1.00 (0.00)**	**1.00 (0.00)**
	f5	**0.60 (0.03)**	**0.00 (0.00)**	**0.00 (0.00)**	**0.00 (0.00)**	-	**0.58 (0.02)**	0.49 (0.97)
	f6	0.51 (0.85)	**0.00 (0.00)**	**0.00 (0.00)**	**0.00 (0.00)**	**0.42 (0.02)**	-	**0.40 (0.02)**
	f7	**0.60 (0.03)**	**0.00 (0.00)**	**0.00 (0.00)**	**0.00 (0.00)**	0.51 (0.97)	**0.60 (0.02)**	-

Lastly, the CCC_TET fitness function was found to detect faults well. With only a few test cases, it was able to kill 91.91%, 99.48%, 94.65%, 99.60%, 58.40%, and 89.58% of the mutants. The significant reduction in execution time in these cases can be attributed to the low number of selected test cases. However, these scores were marginally lower than those obtained from the FDC-based results, and as a result some mutants were missed.

Thus, the FDC-related multi-objective cost-effective fitness functions have proved effective in reducing execution time while obtaining the same mutation detection score as the initial test suite. In some instances, the mutation score was not 100%, but this was a consequence of the inability of the initial test suite to detect all mutations.

Table 7.9: Vargha and Delaney $\hat{A}_{12}$ values and p-values for the fitness functions – NSGAII FRTD

		f1	**f2**	**f3**	**f4**	**f5**	**f6**	**f7**
Enoiu et al.	**f1**	-	**0.95 (0.00)**	**0.92 (0.00)**	**0.00 (0.00)**	0.58 (0.20)	0.55 (0.54)	0.43 (0.26)
	f2	**0.05 (0.00)**	-	0.52 (0.98)	**0.00 (0.00)**	**0.08 (0.00)**	**0.06 (0.00)**	**0.03 (0.00)**
	f3	**0.08 (0.00)**	0.48 (0.98)	-	**0.00 (0.00)**	**0.12 (0.00)**	**0.10 (0.00)**	**0.06 (0.00)**
	f4	**1.00 (0.00)**	**1.00 (0.00)**	**1.00 (0.00)**	-	**1.00 (0.00)**	**1.00 (0.00)**	**1.00 (0.00)**
	f5	0.42 (0.20)	**0.92 (0.00)**	**0.88 (0.00)**	**0.00 (0.00)**	-	0.47 (0.51)	**0.36 (0.02)**
	f6	0.45 (0.54)	**0.94 (0.00)**	**0.90 (0.00)**	**0.00 (0.00)**	0.53 (0.51)	-	0.39 (0.08)
	f7	0.57 (0.26)	**0.97 (0.00)**	**0.94 (0.00)**	**0.00 (0.00)**	**0.64 (0.02)**	0.61 (0.08)	-
Second order	**f1**	-	**0.00 (0.00)**	**0.00 (0.00)**	**0.00 (0.00)**	0.51 (0.52)	0.54 (0.35)	**0.31 (0.00)**
	f2	**1.00 (0.00)**	-	0.50 (1.00)	0.89 (0.25)	**1.00 (0.00)**	**1.00 (0.00)**	**1.00 (0.00)**
	f3	**1.00 (0.00)**	0.50 (1.00)	-	0.89 (0.25)	**1.00 (0.00)**	**1.00 (0.00)**	**1.00 (0.00)**
	f4	**1.00 (0.00)**	0.11 (0.25)	0.11 (0.25)	-	**1.00 (0.00)**	**1.00 (0.00)**	**1.00 (0.00)**
	f5	0.49 (0.52)	**0.00 (0.00)**	**0.00 (0.00)**	**0.00 (0.00)**	-	0.52 (0.77)	**0.30 (0.00)**
	f6	0.46 (0.35)	**0.00 (0.00)**	**0.00 (0.00)**	**0.00 (0.00)**	0.48 (0.77)	-	**0.28 (0.00)**
	f7	**0.69 (0.00)**	**0.00 (0.00)**	**0.00 (0.00)**	**0.00 (0.00)**	**0.70 (0.00)**	**0.72 (0.00)**	-

This might have led to a small increase in execution time because the optimisation algorithm will always attempt to achieve the highest number of detectable failures. One further observation is that the BC_TET metric does not appear to effectively detect faults, as the mutation score reported a significant decrease. Interestingly, CCC_TET was capable of detecting most of the faults with only a few test cases, which indicates fault detection potential. The fitness metrics that obtained maximum mutation score and minimum TET are summarised in Table 7.11.

In total, TET decreased by: 1) 70.80% and 84.14% in the FFTD case, 2) 77.10% and 84.11% in the case of the FFTD, and 3) 89.87% and 89.58% in the FRTD.

Table 7.10: Cost-effective analysis results of the multi-objective search algorithm vs initial test suite

		Enoiu et al.			Second-order		
		TETreduced	**#tc**	**MuScore**	**TETreduced**	**#tc**	**MuScore**
FFTD	**f1**	45.85% (29281ms)	158	100.00%	82.28% (9580ms)	41	99.80%
	f2	98.80% (646ms)	4	78.50%	98.80% (646ms)	4	78.47%
	f3	99.02% (532ms)	3	83.30%	99.02% (532ms)	3	83.30%
	f4	97.46% (1372ms)	7	91.91%	96.90% (1677ms)	6	99.48%
	f5	45.96% (29219ms)	157	100.00%	83.56% (8891ms)	38	99.80%
	f6	45.54% (29447ms)	158	100.00%	80.67% (10452ms)	44	99.80%
	f7	70.80% (15789ms)	83	100.00%	84.14% (8578ms)	36	99.80%
FRTD	**f1**	58.27% (28503ms)	148	100.00%	78.35% (14785ms)	91	99.80%
	f2	98.96% (708ms)	4	80.77%	99.16% (575ms)	4	98.97%
	f3	98.96% (708ms)	4	98.82%	99.18% (558ms)	4	99.04%
	f4	97.02% (2034ms)	8	94.65%	96.61% (2313ms)	9	99.60%
	f5	57.94% (28726ms)	149	100.00%	77.34% (15477ms)	95	99.80%
	f6	58.59% (28281ms)	147	100.00%	79.56% (13962ms)	86	99.80%
	f7	77.10% (15643ms)	74	100.00%	84.11% (10855ms)	63	99.80%
LAUNCHER	**f1**	89.61% (2252ms)	15	80.00%	89.38% (2302ms)	18	95.83%
	f2	99.50% (108ms)	1	39.20%	99.45% (120ms)	1	45.83%
	f3	99.33% (146ms)	1	40.00%	99.44% (122ms)	1	89.92%
	f4	99.02% (213ms)	2	58.40%	99.44% (122ms)	1	89.58%
	f5	89.87% (2195ms)	15	80.00%	89.71% (2230ms)	17	95.83%
	f6	89.18% (2344ms)	16	80.00%	89.78% (2215ms)	17	95.83%
	f7	88.85% (2416ms)	16	80.00%	89.58% (2258ms)	17	95.83%

Table 7.11: Summary of results of the most cost-effective metrics

Use case	**fitness**	**MuScore**	**TETreduced**	**#tc**
FFTD Enoiu et al	FDC_TET_CCC	100.00%	70.80%	83
FFTD second-order	FDC_TET_CCC	99.80%	84.14%	36
FRTD Enoiu et al	FDC_TET_CCC	100.00%	77.10%	74
FRTD second-order	FDC_TET_CCC	99.80%	84.11%	63
LAUNCHER Enoiu et al	FDC_TET_BC	80.00%	89.87%	15
LAUNCHER second-order	FDC_TET_ICC	95.83%	89.78%	16

7.6 Threats to Validity

In this section, we identify threats that could invalidate the evaluation presented above, including threats to internal validity, external validity, and conclusion validity.

7.6.1 Internal Validity

The parameter configurations of the search algorithm could be considered an internal validity concern, because different parameters might give rise to a variance in the results. We selected typical test case selection parameters to minimise this threat. One further concern could also be linked to the use of mutants when calculating FDC and the fitness evaluation function. To mitigate this second issue, we generated two sets of mutants in accordance with the approach of Enoiu et al. [ESv+16] and second-order mutants as reported by [LJB22b].

7.6.2 External Validity

A common external validity issue in software engineering is the generalisation of results. To approximate reality as closely as possible, we used case studies from a real reactor plant. These use cases have been widely reported in the literature for FBD testing [JYCB09, JSC+14, SJB16, SJB18]. In our particular case, three different case studies were selected to ensure a range of internal complexities, and to cover different IEC 61131-3 function groups.

7.6.3 Conclusion Validity

The stochasticity of the search algorithms could be considered a threat to validity from the perspective of confidence in the obtained results. For this reason, each algorithm was executed 50 times to ensure statistical significance and increase confidence in the results.

7.7 Conclusion

The present-day manufacturing industry must respond rapidly and effectively to constantly changing customer demands. In this context, regression testing is commonly employed to reduce time and effort when testing new changes in the manufacturing system. In this chapter, a search-based test selection approach for FBD testing was presented, accomplishing *Objective 3*.

We implemented a multi-objective test selection approach (*Sub-contribution 1*) based on TET, FBD, structural coverage, and FDC. To this end, three coverage criteria (BC, ICC, CCC) were employed as effective criteria (*Sub-contribution 2*), and two sets of mutants were used to obtain the FDC metric, including second-order mutants.

Three use cases were implemented and analysed, including modules from a reactor protection system, with differing levels of complexity and FBD characteristics. The analysis was conducted via 6 experimental scenarios (*Sub-contribution 3*), in which we combined 7 fitness functions. The experiment was carried out twice, as two sets of mutants were used for evaluation. The results show that FDC-based metrics can cost-effectively detect all mutants with respect to the initial test suite. In addition, the combination of CCC criterion and TET metric was found to deliver the best performance, due to its ability to detect faults with few test cases.

As a result, this chapter addresses *Hypothesis 2* as:

Our findings demonstrate that search-based test selection algorithms optimise the selection of the generated test cases in a cost-effective manner, in which the search-based NSGA-II algorithm outperformed RS in 95.24% of the experimental scenarios. Specifically, the NSGA-II algorithm in combination with the CCC_TET metric showed the most optimal solutions in 83.33% of the experimental scenarios.

The following table summarises the contributions, objectives and hypotheses accomplished in this chapter:

Table 7.12: Chapter 7 contributions, objectives and hypotheses

Sub-contributions	**Contributions**	**Objectives**	**Hypotheses**
1, 2, 3	Technical Contribution 2	3	2

Part IV

TIA Portal Testing

Chapter 8

Simulation-based PLC Testing of Siemens PLC Programs

Contents

8.1 Introduction

This chapter presents our methodology to test Siemens PLC programs by using the Siemens PLC simulation tool PLCSim Advanced, in line with *Technical contribution 3*. The solution is based on SiL-based testing of the PLC program, with the use of test oracles. To accomplish this, we fed TIA Portal with the test inputs and oracles obtained in the previous chapters.

The main goal of this chapter is to evaluate the effectiveness of the proposed methodology by applying it to widely used industrial PLCs – in this case Siemens. The results serve as proof of concept for commissioning Siemens PLC software. To achieve this, the following sub-contributions were established:

- **Sub-contribution 1:** Approach to automate SiL testing in TIA Portal with the use of PLCSim Advanced.
- **Sub-contribution 2:** Generation of TIA Portal Test Suite application tests based on test oracles generated in Chapter 6.
- **Sub-contribution 3:** PLC setup and OB networks definition for the monitoring of scan cycles and synchronisation of the program.

The presented sub-contributions are directly linked to the contributions, objectives, and hypotheses described in Table 8.9:

Table 8.1: Chapter 8 contributions, objectives and hypotheses

Sub-contributions	Contributions	Objectives	Hypotheses
1, 2, 3	Technical Contribution 3	1, 4	3

To this end, the main objectives of this chapter are stated below:

Objective 1: Build an interoperable framework than enables the continuous commissioning of PLCs.
Objective 4: Develop and evaluate a methodology to automatically perform SiL-based commissioning on industrial PLCs.

Section 8.2 describes our testing methodology and the key steps involved in the process (Sub-contribution 1). Specifically, test oracles retrieved from Chapter 6 were used to define assertions in TIA Portal Test suite (Sub-contribution 2) for benchmarking against the simulation results. In addition, the required PLC setup, data type and OB definitions were defined for Siemens 1500 PLC families (Sub-contribution 3). Section 8.3 details the empirical evaluation of the efficacy of the oracles to validate industrial PLC programs with 5 experimental scenarios. The results are reported in Section 8.3.2, and are further discussed in Section 8.3.3. Threats to validity are outlined in Section 8.3.4. Finally, Section 8.4 presents the conclusions and future lines of the presented study.

8.2 PLC Testing with the Use of PLCSim Advanced

In this section, we present a methodology to test Siemens-specific PLC programs. This method involves using the test cases and test oracles generated in Chapter 6, by transferring this knowledge to Siemens TIA Portal – a fully integrated automation software for configuring, programming, testing, and diagnosing Siemens controllers. Figure 8.1 depicts a simplified overview of the methodology presented in Section 5.3, focusing mainly on the key steps involved in this chapter. It is important to note that this chapter aims to address *Hypothesis 4*. As a result, our main focus is on validating the test oracles by executing all the test cases generated in Chapter 6. Test selection, therefore, becomes an optional step in this chapter and has been excluded to ensure the validation of all test oracles.

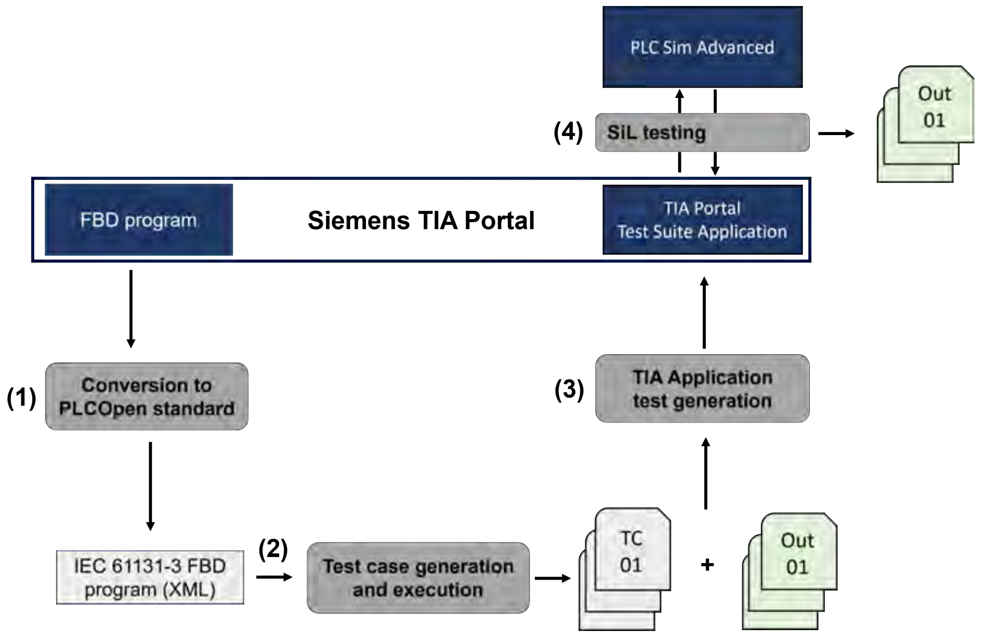

PLC Sim Advanced
(4)
SiL testing
Out 01
FBD program
Siemens TIA Portal
TIA Portal Test Suite Application
(1)
Conversion to PLCOpen standard
(3)
TIA Application test generation
IEC 61131-3 FBD program (XML)
(2)
Test case generation and execution
TC 01
+
Out 01

Figure 8.1: TIA Portal testing overview

As a first step, it is necessary to export the Siemens PLC program and to convert it into the PLCopen XML format (Step 1). Next, the techniques employed in Chapter 6 are used to generate the tests and retrieve the test oracles (Step 2). These are required to define application test cases for TIA Portal Test Suite, in which test cases are used to specify input values, and test oracles assert output values (Step 3). These application tests are then imported to the TIA Portal Test Suite. In the final step, Siemens PLC programs are tested by executing the Test Suite against PLCSim Advanced (Step 4).

The key steps involved in the testing process are detailed in the following sections. Section 8.2.1 details Siemens PLC software main characteristics. Section 8.2.2 sets out PLC program standardisation alternatives, and describes the solution applied in this approach. Sections 8.2.3-8.2.4 explain the process for generating and executing TIA Test Suite application tests. Finally, the requirements in terms of PLC configuration, data type definitions and OBs are described in Section 8.2.5.

8.2.1 Siemens PLC Software

Siemens is a leading manufacturer of industrial automation equipment. Their PLCs are based on a version of the IEC 61131-3 standard for programmable controllers called STEP 7.

Siemens PLCs support a number of programming languages specified in the IEC 61131-3 standard to some extent. These include LD, ST, FBD, and SFC. In addition, they include additional proprietary languages such as Structured Control Language (SCL), and Assembly Language for STEP 7 (AWL):

- Siemens SCL: a high-level, textual programming language specific to Siemens PLCs. It is used to write programs in a structured, high-level format, and is similar to the C programming language.
- Siemens AWL: a low-level, textual programming language specific to Siemens PLCs. It is used to write programs in a machine-readable format, and is similar to assembly language.

In a Siemens PLC program, the software blocks are divided into the following units:

- Organisational Block (OB): a sequence of instructions used to control the flow of execution of a PLC program. This is the starting point for the program, similar to the main function in software engineering, and determines the order of the instructions to be executed.
- Function Call (FC): a block of instructions that perform a specific task. An FC block consists of executable code with inputs, outputs, and temporary variables. These variables are deleted from memory after the execution of the function.

- Function Block (FB): a block of instructions that perform a specific task, similar to a FC. The only difference is that a FB can also have internal variables to store intermediate results, in addition to inputs, outputs, and temporary variables. These results are stored in instance data blocks, and remain in the memory after the execution of the FB.

- Data Block (DB): a block of memory that stores data in a PLC program. Static data is typically stored in global DBs, which can be accessed by any instruction in the program.

In this chapter, we present a methodology to test Siemens FBD programs, based on PLCopen XML standard. The employed software blocks comprise, FCs and/or FBs that contain specific task functions.

8.2.2 PLC Code Standardisation

A method for generating a standard format of the PLC code should be developed. This is a crucial step as it ensures the user can integrate PLC software programs from different vendors, libraries, and projects into the various development environments. In the present study, we followed the PLCopen group committee guidelines, which establish an open interface that supports a broad set of software tools. In particular, we employed TC6 for XML workgroup, as it facilitates the transmission of screen-based data to other platforms [SWT12].

Siemens PLCs can import and export program files in XML format, which can be used to transfer programs in Siemens-specific products. However, the XML format is not directly compliant with the PLCopen XML standard, which is an open standard designed to exchange IEC 61131-3 projects. Hence, XML programs may not be compatible with other PLCs or programming tools that do not support this format.

Another solution is to export STL/AWL files. However, converting FBD/LAD to STL is only possible in S7-300 and S7-400 PLC's in TIA Portal. The newer PLCs S7-1200/1500 can only switch between LAD and FBD.

According to Siemens STEP 7 standards compliance manual [Sie15], the instruction list AWL/STL corresponds to IEC 61131-3 language STL/IL. Nonetheless, there exist some discrepancies in the command syntax – the syntax of some operators is different – which results in incompatibility in project exchanges. In addition, Siemens AWL/STL includes more commands than the IEC 61131-3 standard IL language.

The CODESYS Development System is the IEC 61131-3 programming tool for industrial control and automation technologies, and complies with the PLCopen XML standard. As a result, PLC programs developed in Codesys can be exported in an XML scheme for IEC 61131-3 languages, as they are platform-independent.

Table 8.2 summarises the limitations of the aforementioned solutions to obtain PLCopen XML programs.

Table 8.2: PLCopen XML standardisation alternatives and limitations

Option	Method	Limitations
1	Exporting Siemens code in XML format	Siemens specific XML format, not compliant with the PLCopen XML, and requires a conversion tool
2	Exporting Siemens code in AWL/STL format	Not fully compliant with the IEC 61131-3 IL language, owing to some code syntax alterations, and requires mapping of commands
3	Exporting Codesys code in XML format	Compliant with the PLCopen XML, but no means exist to import Siemens code into codesys without any intermediate conversion tool

Given that Codesys is the only development environment that is fully compliant with the IEC 61131-3 standard, we selected this tool to generate PLCopen XML code. As there are no means of importing Siemens-specific PLC code into Codesys, we manually replicated the FBD blocks in Codesys, as illustrated in Figure 8.2. Nevertheless, the result serves as a proof-of-concept to generate test cases based on PLCopen XML files.

Although Codesys is compliant with the PLCopen XML guidelines, the following requirements must be satisfied to import the XML file into the test generation tools. At this stage, we manually made the modifications indicated below, however, we envisage automating these formatting steps by integrating them into the FBDTester 3.0 and MuFBDTester 3.0.

- Project, fileHeader and conctentHeader should be amended.
- Extra zeros should be removed from *block:localID* and *connection:refLocalID*.
- Formal parameters should be capitalised.
- Special characters (e.g. #, “, ’) should be avoided.
- The naming of the formal parameters should match the naming used in the calculation and function libraries.

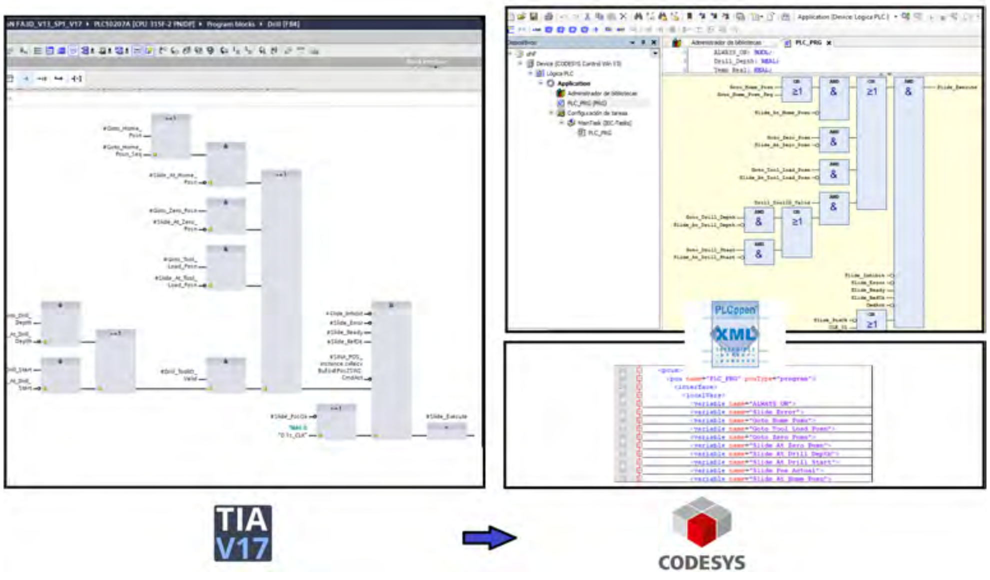


Figure 8.2: PLCopen XML conversion via Codesys

8.2.3 Generation of Test Suite Application Tests

As described in Chapter 6, we evaluated the output values of the test cases in every intermediate cycle. In this section, we detail how those values were utilised to formulate test assertions, which are then employed to test and validate the FBD program in the TIA Portal.

TIA Test Suite is an add-on feature of TIA Portal V17, and defines style guide rules and application tests. The latter functionality is of particular interest as it can create application test cases that verify the output of the PLC program. This is accomplished by setting the values of input variables and verifying the expected outcome after a specific number of scan cycles or a dedicated timespan. Assertion statements are used to compare the current result with the expected values, and the results of which determine failure or success.

In this study, we generated application tests to benchmark the output values of each execution run with the results obtained in Chapter 6, which serve as test oracles. This was achieved via the following steps:

1. **Scope selection**: subject program PLC name.
2. **Variable definition**: definition of PLC input and output variable tags under test. These variables are of boolean or integer array data type. To define and assert values for each scan cycle, multiple variables were defined for

each input/output array.

3. **Statement definition**: test statements are used to assign values to PLC program block variables, define output assertions, and specify the number of scan cycles or timespan to be used during execution.

   a) **Variable definition**: we defined input values from the test cases generated in Chapter 6. Those values were defined for each scan cycle of the input variables.

   b) **Execution run statement**: period of time or number of runs to be executed. We specified the time span in this case, which is a multiple of the scan cycle time. Given that the application test requires 4 scan cycles to run, we defined $\_t0$ as the first scan cycle. Hence, the execution run was defined as the minimum cycle time of the PLC, e.g. $50ms$. Similarly, as $\_t3$ was the last iteration, we defined the execution run as $200ms$.

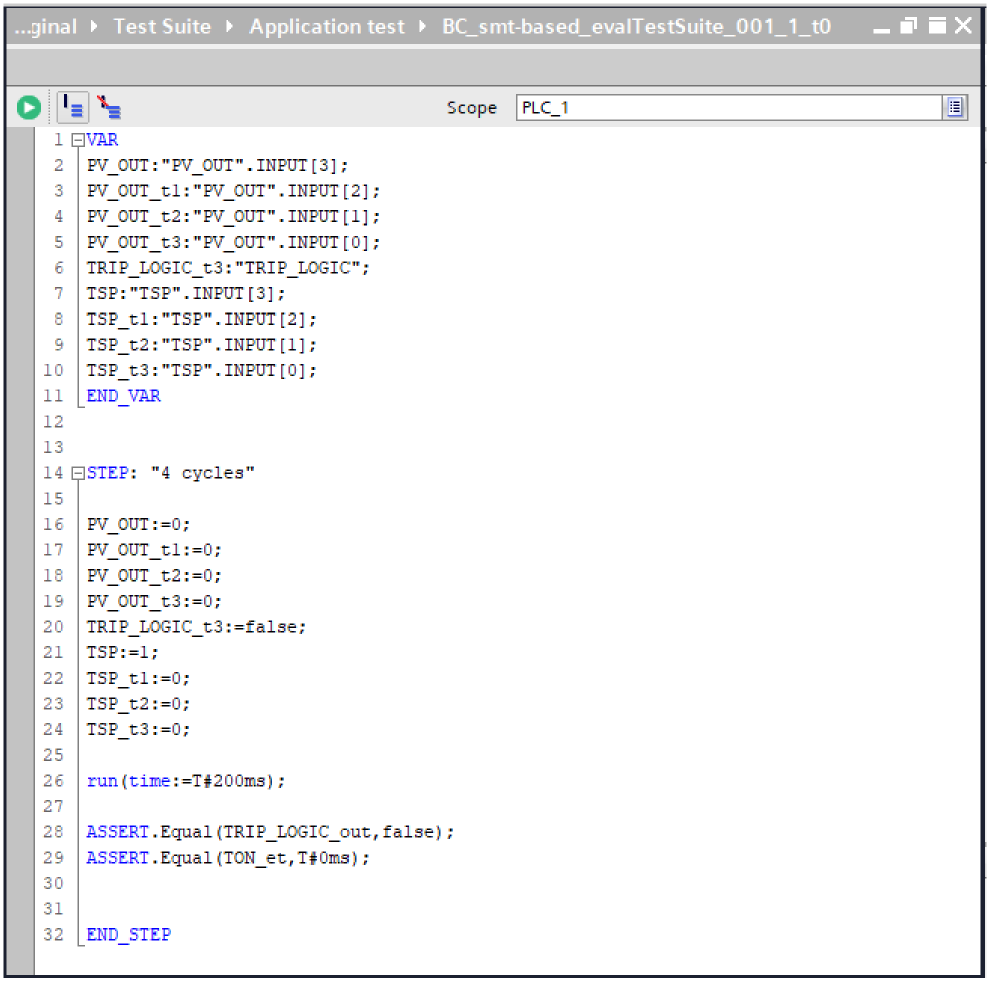
   c) **Assertion statement**: we defined expected output values based on the test execution results obtained in Chapter 6. These assertions were defined for each scan cycle of the output variables.

We generated an application test for every iteration of the PLC, in which the resulting outcome was asserted after the specified time span. A converter (*TIAtestSuiteConverter.java* presented in Appendix C) was developed in Java to generate TIA Test Suite application tests based on the test cases and test oracles retrieved in Chapter 6. Figure 8.3 depicts an example application test case generated by the converter. The test case was specifically designed to test the output of the variables of the PLC_1 after four iterations. The minimum number of scan cycles is, however, calculated based on characteristics of the FBD program, as indicated in Section 6.3.4.

The next step was to integrate the generated test cases into TIA Portal test suite. This was automatically performed with the *import test cases* option of the Test Suite.

8.2.4 Test Suite Execution

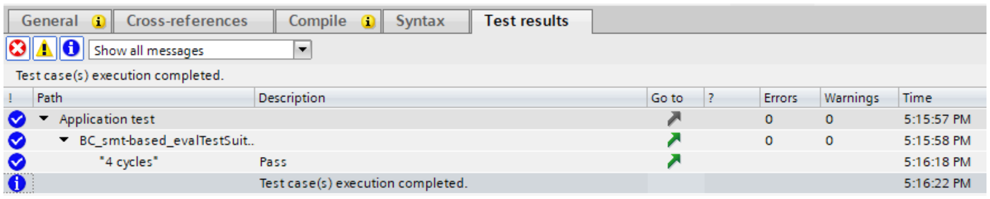
To execute the test cases in the FBD program, TIA Portal automatically creates a S7-PLCSim Advanced instance – a virtual replica of the hardware in place – and establishes a connection with it. TIA Portal Test Suite then compiles the program and transfers the hardware configuration to the PLCSim Advanced instance. All variables are also loaded onto the device. Once successfully completed, application tests are


...ginal ▸ Test Suite ▸ Application test ▸ BC_smt-based_evalTestSuite_001_1_t0
Scope PLC_1

```
VAR
PV_OUT:"PV_OUT".INPUT[3];
PV_OUT_t1:"PV_OUT".INPUT[2];
PV_OUT_t2:"PV_OUT".INPUT[1];
PV_OUT_t3:"PV_OUT".INPUT[0];
TRIP_LOGIC_t3:"TRIP_LOGIC";
TSP:"TSP".INPUT[3];
TSP_t1:"TSP".INPUT[2];
TSP_t2:"TSP".INPUT[1];
TSP_t3:"TSP".INPUT[0];
END_VAR


STEP: "4 cycles"

PV_OUT:=0;
PV_OUT_t1:=0;
PV_OUT_t2:=0;
PV_OUT_t3:=0;
TRIP_LOGIC_t3:=false;
TSP:=1;
TSP_t1:=0;
TSP_t2:=0;
TSP_t3:=0;

run(time:=T#200ms);

ASSERT.Equal(TRIP_LOGIC_out,false);
ASSERT.Equal(TON_et,T#0ms);


END_STEP
```

Figure 8.3: Example of a TIA Test Suite application test case

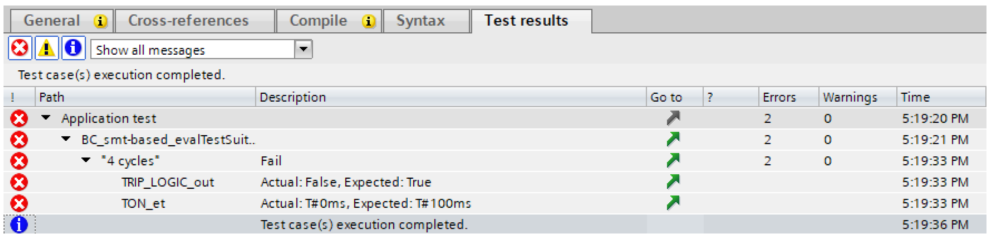
executed and asserted one by one. Tests are passed if the asserted output value is the same as the expected result. If not, a test failure is reported indicating the actual value, as seen in Figures 8.4-8.5.



Test case(s) execution completed.

!	Path	Description	Go to	?	Errors	Warnings	Time
✓	Application test		↗		0	0	5:15:57 PM
✓	BC_smt-based_evalTestSuit..		↗		0	0	5:15:58 PM
✓	"4 cycles"	Pass	↗				5:16:18 PM
i		Test case(s) execution completed.					5:16:22 PM

Figure 8.4: Successful test execution



Test case(s) execution completed.

!	Path	Description	Go to	?	Errors	Warnings	Time
✗	Application test		↗		2	0	5:19:20 PM
✗	BC_smt-based_evalTestSuit..		↗		2	0	5:19:21 PM
✗	"4 cycles"	Fail	↗		2	0	5:19:33 PM
✗	TRIP_LOGIC_out	Actual: False, Expected: True	↗				5:19:33 PM
✗	TON_et	Actual: T#0ms, Expected: T#100ms	↗				5:19:33 PM
ℹ		Test case(s) execution completed.					5:19:36 PM

Figure 8.5: Unsuccessful test execution

8.2.5 PLC Setup Requirements

This section outlines the required PLC configuration setup to conduct the tests with TIA Test Suite, and details the necessary changes in the definition of data types and main organisational blocks of the PLC to track the values for each scan cycle.

PLC Configuration

It is important to note that TIA Test Suite is only available in the STEP 7 Professional V17 version of TIA Portal. In addition, the use of application tests requires PLCSim Advanced V3 or V4 to be installed.

With regard to the PLC requirements, Test Suite is currently only supported for 1500 PLC families. Moreover, the configuration option should be enabled to support the simulation during block compilation. This option allows test cases to be executed using PLCSim Advanced.

Another important configuration aspect is the scan cycle time. To ensure the same execution time for all cycles, the minimum and maximum scan cycles should be the same, or marginally equal.

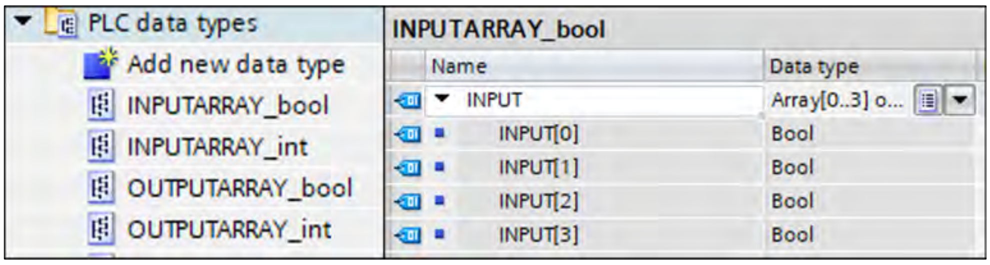
Definition of Array Data Types

Array data types were created to track variable values in every scan cycle. Specifically, input array data types were defined to write values on input variables, whilst output array data types were created to read values from output variables. Figure 8.6 illustrates the integer and boolean array data types that were used for input and output variables.

All input and output variables were thus mapped to arrays of the same data types, i.e. an input variable of boolean type was mapped to a boolean input array data type.

Definition of Main OB1 Networks

OB blocks are the main organisation or program blocks that determine the structure of a PLC program. In this case, we used OB1 to cyclically call the FBD under test



PLC data types
- Add new data type
- INPUTARRAY_bool
- INPUTARRAY_int
- OUTPUTARRAY_bool
- OUTPUTARRAY_int

INPUTARRAY_bool

Name	Data type
INPUT	Array[0..3] o...
INPUT[0]	Bool
INPUT[1]	Bool
INPUT[2]	Bool
INPUT[3]	Bool

Figure 8.6: Array data types

when conducting unit testing with the Test Suite application tests. However, some additional networks were required to track the scan cycle values, and synchronise the program accordingly.

Network 1: First scan A first scan network was designed to reset memory tag variables (particularly those employed to track the number of iterations and cycles) in the first scan cycle (Figure 8.7). To this end, we enabled the system memory bits from the *System and Clock memory* properties, and used the memory bit allocated to the first scan cycle, i.e. *M1.0*. This memory bit is high on the first cycle and remains low for the remaining cycles.

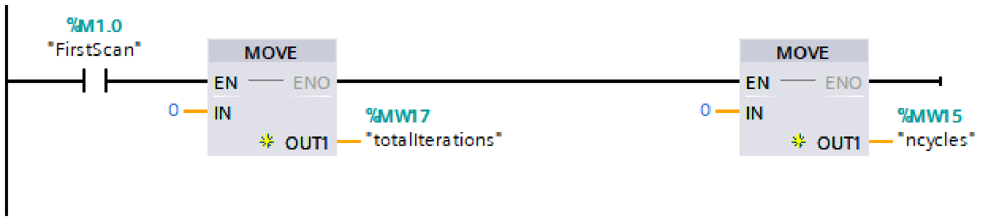


Figure 8.7: First scan network

Network 2: Get array size Since FBs require a minimum number of scan cycles to activate outputs, we developed a means to track the cycles, and execute the respective variable values.

Here, a new FB network was generated in the main OB block, which calculates the size of the array to be used in the FBD program, as indicated in Figure 8.8. This is used to define the total number of iterations required in the program under test. The FBD block receives as an input the array variable to be read, and the tag variable $totalIterations$, as the name says, saves the total number of iterations in the internal memory.

In this way, a function was defined within the FBD to count the number of elements in the array, which corresponds to the number of cycles. We subtracted one cycle,

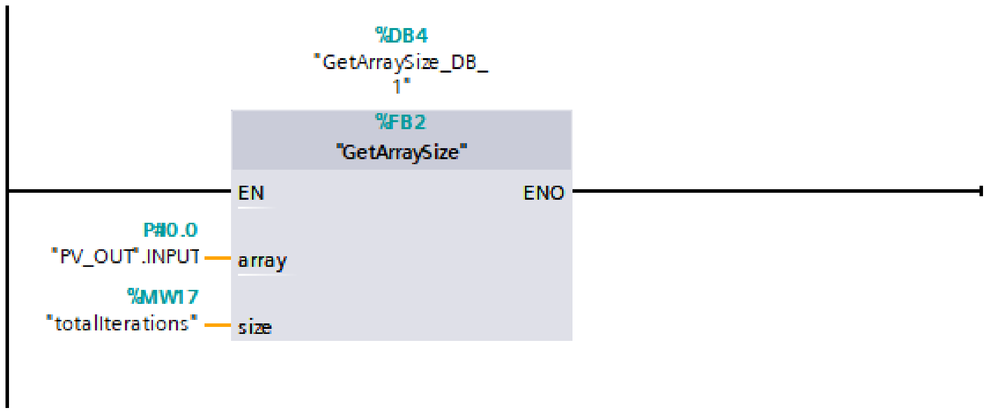


Figure 8.8: Get array size network

however, as the iteration numbers that we defined during the test generation process start counting from 0.

Network 3: Feedback loop Closed-loop control systems are used to update the state of an input based on the system output without human intervention. Hence, we designed a new FB network to keep track of the feedback values for each scan cycle, as indicated in Figure 8.9. This block receives as input the current iteration number $nIteration$ and updates the value of feedback variables, i.e. $TRIP\_LOGIC$ in this case.

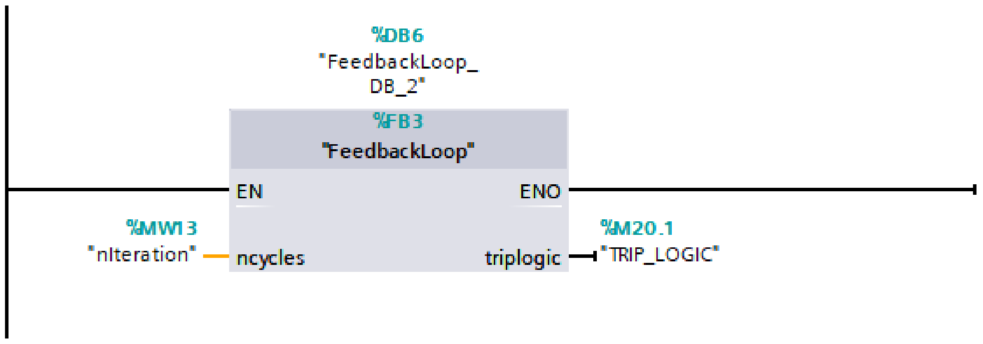


Figure 8.9: Feedback loop network

Network 4: Reset A reset network might be necessary to force the reset of internal variables of some IEC 61131-3 FBs such as counters or timers, since the execution of the program is started after the first scan cycle (one cycle of delay). As depicted in Figure 8.10, we created a function to reset a timer on-delay FB to synchronise with the scan cycles when testing the FBD program. This ensured that the timer block internal variable *IN*, which is used to activate a timer, is set to FALSE in the first scan cycle.

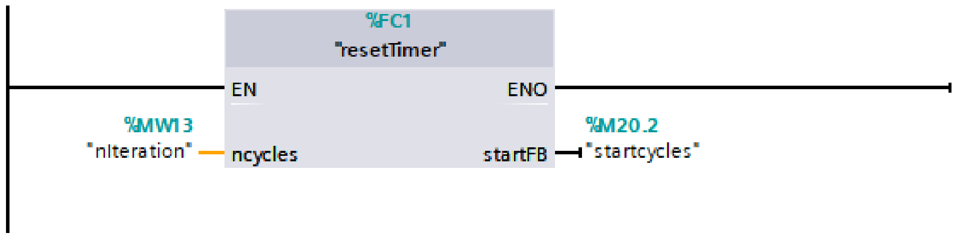


Figure 8.10: Reset network

Network 5: FBD unit testing We generated an FBD network to manage the iteration numbers for each scan cycle when testing the subject FBD program, as illustrated in Figure 8.11. This consists of: 1) a *Greater than or Equal to FC* that checks if the maximum number of iterations was reached, 2) an *addition FC* to count the number of cycles, and 3) a *subtraction FC* that subtracts one cycle from the number of cycles, returning the current iteration number.

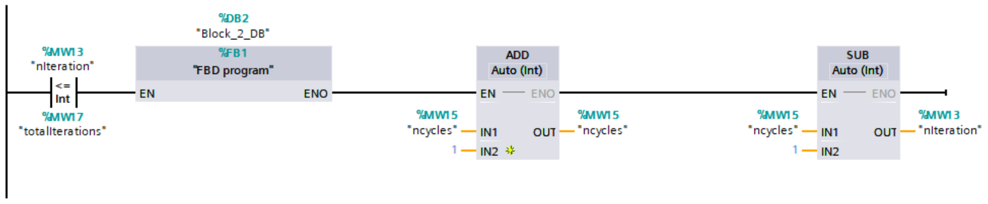


Figure 8.11: Main network for FBD unit testing

8.3 Evaluation

8.3.1 Case Studies

The case studies were conducted in real industrial PLC programs. We selected five FBD networks to perform unit testing, three of which were used at Danobatgroup for their machining solutions, and two were employed in the Omnifactory case study at UNOTT. Specifically, we analysed a Robot FBD, Safety FBD, and Gantry FBD at Danobatgroup, while a two network FBD that controls a drilling machine was tested in the Omnifactory case.

In addition, time-dependent FBDs were tested by a simplified FBD program used by the reactor protection system developed by the KNICS project. These use cases are detailed in the following subsections.

Use case 1: Safety FBD

The use of signals with negated logic is widespread in safety control systems, and particularly in those which feature switches and inductive or capacitative NPN sensors (i.e., Negative-Positive-Negative sensors that provide an active low output). Implementation errors related to normally open/closed contacts can give rise to errors of particular significance in the subsequent code of the PLC program, especially in safety-related systems. Figure 8.12 depicts a safety ladder diagram network used at Danobatgroup, which consists of 8 normally open contacts, 2 normally closed contacts, and a bistable set-reset FB.

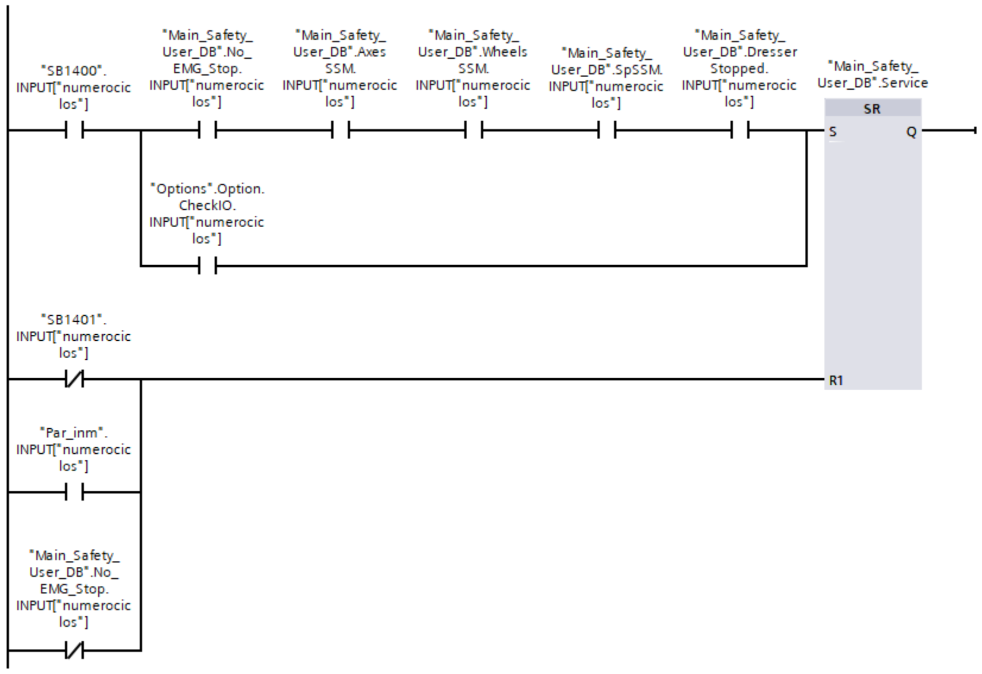

"SB1400". INPUT["numerociclos"]
"Main_Safety_User_DB".No_EMG_Stop. INPUT["numerociclos"]
"Main_Safety_User_DB".Axes SSM. INPUT["numerociclos"]
"Main_Safety_User_DB".Wheels SSM. INPUT["numerociclos"]
"Main_Safety_User_DB".SpSSM. INPUT["numerociclos"]
"Main_Safety_User_DB".Dresser Stopped. INPUT["numerociclos"]
"Main_Safety_User_DB".Service
SR
S
Q
R1
"Options".Option. CheckIO. INPUT["numerociclos"]
"SB1401". INPUT["numerociclos"]
"Par_inm". INPUT["numerociclos"]
"Main_Safety_User_DB".No_EMG_Stop. INPUT["numerociclos"]

Figure 8.12: Safety FBD

Use case 2: Gantry FBD

A gantry is typically an external machine and for this reason, it is crucial to ensure that the signal communication between the gantry and the PLC is correct. The Gantry network employed in this case comprises 11 AND logic FCs, and 6 OR logic FCs, which is too large to illustrate here.

Use case 3: Robot FBD

Industrial robots are also external devices commonly used on the shop floor. The PLC should therefore guarantee that the orchestration of these devices is correct, in a similar fashion to the gantry. Our robot FBD consists of 7 normally open contacts, 3 normally closed contacts, a bistable set-reset FB, and a negative edge trigger FB (Figure 8.13).

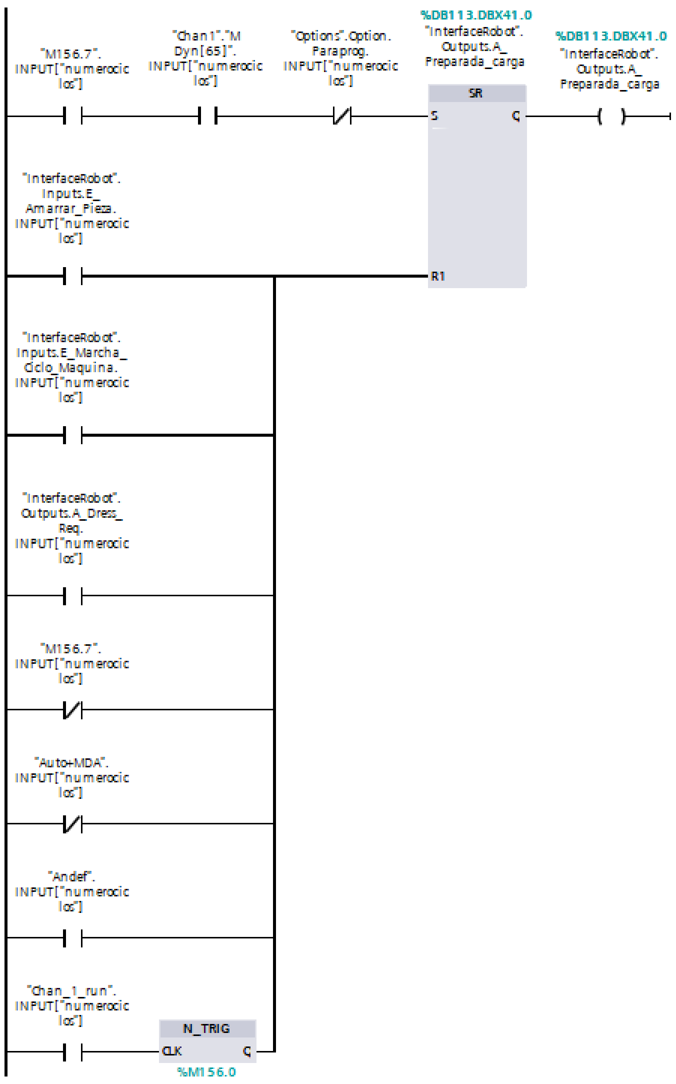


Figure 8.13: Robot FBD

Use case 4: Drilling FBD

In this case, we employed two FBD networks from the drilling FBD to assess the testing capability with multiple networks. The former is used to set drilling position flags, whereas the latter activates the execute bit when a position flag is high. The FBD is composed of 13 AND logic FCs, 4 OR logic FCs, 5 comparison Equal FCs, and contains a feedback loop. Commercial in confidence precludes us from illustrating this FBD in this document.

Use case 5: simTRIP

The main reason for deploying this use case was to test time-related FBs in the simulation environment, since the other use cases do not include such function blocks. These were specifically employed to send a shutdown signal to a nuclear reactor plant under given input conditions after a pulse time delay, as described in Section 5.4.3.

Main Characteristics of the Use Cases

The main characteristics of the aforementioned use cases are summarised in Table 8.3, including the total number of inputs, outputs and blocks, the IEC 61131-3 function groups utilised, and the inclusion of feedback loops. It should be noted that use cases 1-3 were converted from ladder logic to FBD language using the TIA Portal *switch programming language* feature. As a result, normally open and closed contacts were mapped to IEC 61131-3 OR and AND logical FCs, based on their series-parallel connection.

Table 8.3: FBD characteristics of the use cases

	#inputs	**#outputs**	**#blocks**	**IEC 61131-3 function groups**	**Feedback**
Use case 1	10	1	5	logic, bistable	
Use case 2	26	8	17	logic	
Use case 3	10	1	4	logic, bistable, trigger edge	
Use case 4	22	6	22	logic, comparison	X
Use case 5	3	2	3	logic, comparison, timer	X

Research Questions

The objective of the experiments is to evaluate the effectiveness of the test oracles in validating FBD networks with TIA Test Suite. To this end, we defined the following research question, which aims to address *Hypothesis 4*:

RQ11. Can the generated test oracles be used to test the Siemens PLC program using the TIA Portal Test Suite?

Metrics

The evaluation was carried out by measuring the percentage of passed and failed tests after the execution of the tests. Performance was evaluated in terms of CPU execution time (in minutes) when executing application tests against PLCSim Advanced. Execution time and fail/success description of each application test are reported automatically within the *Test results* tab in TIA Test Suite.

Experimental Setup

Each use case was executed in TIA Portal V17 and evaluated against PLCSim Advanced 3.0. The PLC used was a CPU 1512C-1 PN configured with a minimum cycle time of 50 ms and maximum cycle time of 51 ms. The remaining configurations were left as default.

In total, we defined 5 experiments to analyse the RQ11, one experiment per use case. Table 8.4 details the OB1 networks used in each. Networks 1, 2 and 5 are required for all the use cases, in which network 1 restarts memory tags in the first cycle, network 2 obtains the number of iterations required to test, and network 5 is the main network that tests the subject PLC program. In addition, the internal feedback loops in use case 4 require the use of network 3. Lastly, use case 5 includes all networks, as it consists of a timer block that needs to be synchronised with the initial iteration.

Table 8.4: Networks used in each of the use cases

	Use case 1	**Use case 2**	**Use case 3**	**Use case 4**	**Use case 5**
Network 1	X	X	X	X	X
Network 2	X	X	X	X	X
Network 3				X	X
Network 4					X
Network 5	X	X	X	X	X

The use case 2 was executed once, as the FBD program consists only of FCs, rather than FBs. In other words, there are no internal memory states and no dependencies related to previous iteration values. In contrast, use case 1 includes a bistable FB, use case 3 comprises bistable and edge detection FBs, and use case 5 also relies on internal states due to the timer FB. In addition, use case 4 features some internal dependencies due to the feedback loops. These use cases were therefore executed 50 times for evaluation purposes, in which the results were evaluated with statistical tests.

8.3.2 Results

To address the RQ11 we first generated application test cases by following the steps presented in Section 8.2. This included the conversion of PLC programs to XML code through codesys, generation of coverage-based and mutant-based test cases, and execution of tests to obtain test oracles. Table 8.5 sets out the application test cases that were automatically generated for TIA Test Suite in each use case. This comprises BC, ICC, and CCC criteria-based test cases, and mutation-based test cases.

Table 8.5: Test cases generated for each of the use cases

	#BC TCs	**#ICC TCs**	**#CCC TCs**	**#Mutation TCs**
Use case 1	2	3	3	29
Use case 2	1	2	2	43
Use case 3	2	3	3	24
Use case 4	2	5	5	122
Use case 5	1	1	2	16

The execution results of the test suite are shown in Table 8.6 for use case 2, and Table 8.7 for use cases 1, 3, 4 and 5. The latter sets out the mean (μ) values of the success rate and TET over the 50 execution runs.

Table 8.6: Test Suite execution results

	#total TCs	**SuccessRate**	**TET**
Use case 2	48	100.00%	73 min

Table 8.7: Test Suite execution mean values of 50 runs

	#total TCs	**μ SuccessRate**	**μ TET**
Use case 1	37	100.00%	24 min
Use case 3	32	100.00%	41 min
Use case 4	134	100.00%	132 min
Use case 5	20	85.25%	21 min

Uses cases 1, 2, 3 and 4 were successfully tested, which means that the simulation test outputs matched the results obtained in the FBDTester 3.0 (i.e. test oracles). However, the outcome of use case 5 differed from the expected results, achieving an average success rate of 85.25%.

With regard to execution time, the TET of the use cases significantly increased with the number of test cases, as well as the complexity of the subject programs (i.e. the number of inputs, outputs and blocks). TET was 73 min for use case 2. For use cases 1, 3, 4 and 5, a mean TET of 24 min, 41 min, 132 min and 21 min was achieved respectively over 50 execution runs. In this regard, Figure 8.14 details the relationship between TET and the number of test cases in *subplot a*, and TET and the number of blocks in *subplot b*. In both cases, TET is higher with a higher number of blocks and test cases.

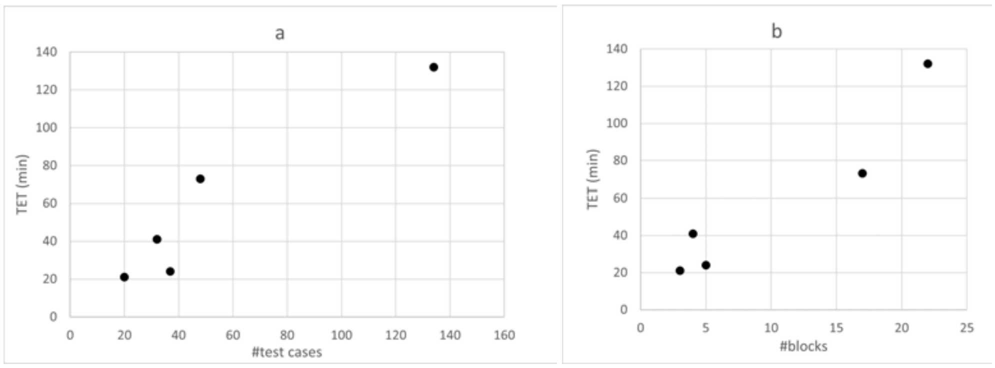


Figure 8.14: TET results a) TET vs number of test cases b) TET vs number of blocks

Figure 8.8 sets out the test results of the use case 5, in which the mean values over the 50 execution runs are shown. The achieved mean success rate at each iteration cycle of the test cases is reported, with a maximum of 4 cycles. Test case TET is also detailed. The results show that during the first scan cycle, all the tests achieved a success rate of 100%. However, this score decreased for 30% of the test cases during the second cycle, 40% of the test cases in the third cycle, and 50% of the test cases during the fourth cycle.

Table 8.8: Application test case mean values of 50 runs

	μ **successRate**				μ **TET**
Application test	**1st cycle**	**2nd cycle**	**3rd cycle**	**4th cycle**	**All cycles**
BC_smt-based_evalTestSuite_001_1	100.00%	46.00%	58.00%	74.00%	69 sec
ICC_smt-based_evalTestSuite_001_1	100.00%	100.00%	44.00%	66.00%	63 sec
CCC_smt-based_evalTestSuite_001_1	100.00%	100.00%	44.00%	58.00%	63 sec
CCC_smt-based_evalTestSuite_001_2	100.00%	100.00%	100.00%	100.00%	64 sec
mutation_0000_test	100.00%	46.00%	48.00%	44.00%	65 sec
mutation_0001_test	100.00%	100.00%	100.00%	100.00%	64 sec
mutation_0002_test	100.00%	100.00%	100.00%	38.00%	65 sec
mutation_0003_test	100.00%	100.00%	100.00%	100.00%	65 sec
mutation_0004_test	100.00%	100.00%	100.00%	100.00%	65 sec
mutation_0005_test	100.00%	100.00%	100.00%	28.00%	65 sec
mutation_0006_test	100.00%	100.00%	100.00%	100.00%	65 sec
mutation_0007_test	100.00%	100.00%	100.00%	100.00%	65 sec
mutation_0008_test	100.00%	40.00%	48.00%	54.00%	66 sec
mutation_0009_test	100.00%	50.00%	66.00%	52.00%	66 sec
mutation_0010_test	100.00%	100.00%	100.00%	100.00%	66 sec
mutation_0011_test	100.00%	100.00%	100.00%	52.00%	65 sec
mutation_0012_test	100.00%	100.00%	100.00%	100.00%	65 sec
mutation_0013_test	100.00%	100.00%	100.00%	100.00%	64 sec
mutation_0014_test	100.00%	44.00%	76.00%	100.00%	64 sec
mutation_0015_test	100.00%	40.00%	44.00%	60.00%	65 sec

8.3.3 Discussion

The results indicate that the generated test oracles are valid for testing use cases 1-4. This demonstrates that our methodology can effectively test IEC 61131-3 AND and OR logic functions, equality functions, set-reset bistable elements, and negative edge triggers.

Moreover, the presented testing methodology is also valid for testing feedback loops, as shown in use case 4. Feedback loops rely on the outputs of the previous scan cycles, in which values are stored in the internal memory.

In contrast, use case 5 did not achieve the expected outcomes. Despite the test cases being successfully tested in the first cycle, the success rate decreased over the following cycles. Interestingly, this decrease was higher with a higher number of cycles. This is likely because we used a timer FB in the FBD, in which a precise scan cycle is required to perform the tests. However, the scan cycle varied by 1-2 ms in some of the executions, meaning that the timer was not activated on the right cycle. As a result, the test inputs were not synchronised with the scan cycles, and the results were not aligned accordingly. Therefore, we can conclude that our oracles are not valid for testing time-related FBs, such as TON, and this remains an open issue to solve in future lines.

As regards TET, the simulation results revealed a significant increase in execution time compared to Java. The process of test case execution in TIA Portal comprises various steps, resulting in a higher execution time. Firstly, it automatically creates an instance of PLCSim Advanced. It compiles the selected PLC project, and if successful, downloads the PLC configuration to the PLC simulation instance for executing the generated test cases. The process is similar to the execution on a real PLC hardware. TIA Portal communicates with PLCsim Advanced (instead of a real PLC) to execute each test case sequentially. TIA Portal sends the inputs specified in the Test Suite application test to the simulated PLC instance, which processes them according to the subject program logic. Afterwards, TIA Portal deletes the instance, generates a detailed summary of the results, and saves them as log files. All these steps have an impact on the execution time.

In addition, the results in Figure 8.14 disclosed that other factors might also increase TET. Overall, a higher number of test cases led to a longer execution time. On the other hand, there appears to be a trend where TET increases with a higher number of FC/FB blocks. However, these trends do not seem to apply when benchmarking *Use case 1* and *Use case 3*, in which *Use case 3* had a higher TET despite having fewer test cases and blocks. One hypothesis could be that there are other internal factors that introduce additional overhead. For example, FBs may have dependencies on internal memory states or previous iteration values. In the case of *Use case 3*, there was a trigger edge detection FB, which could require some extra time. Nevertheless, this justifies the need to optimise the selection of the test cases. Therefore, the test case selection approach presented in Chapter 7 could be applied to execute only a subset of the test cases and reduce execution costs.

8.3.4 Threats to Validity

In this section, we identify threats that could potentially invalidate the evaluation presented above, including threats to external and conclusion validity.

8.3.5 Conclusion Validity

The obtained results could be compromised by the accuracy of the scan cycle. Despite defining a minimum scan cycle of 50 ms, the scan cycle variations might lead to different results. Hence, FBDs that rely on previous cycles or internal states were executed 50 times to improve confidence in the results.

8.3.6 External Validity

The generalisation of results could be considered an external validity issue. To this end, we employed 5 industrial use cases from Danobatgroup and the University of Nottingham to test real industrial scenarios.

On the other hand, the generalisation of results could also be affected by the various IEC 61131-3 functions. In the present study logic functions, comparison functions, bistable, edge-trigger, and timers were tested. The presented results are therefore limited solely to these IEC 61131-3 FCs and FBs, which does not include arithmetic and non-linear arithmetic functions, numerical functions, selection functions, and counters.

8.4 Conclusion

In this chapter, we have presented a simulation-based testing methodology (*Sub-contribution 1*) to validate the logic of Siemens PLC programs with the use of TIA Portal Test Suite. To accomplish this, we defined all necessary changes in the PLC configuration and predefined 5 types of networks to be used in the main OB of subject PLC programs (*Sub-contribution 3*). As a result, TIA Portal Test Suite application tests were defined with the test cases generated in previous chapters. Test outputs were then benchmarked with the test oracles obtained in Chapter 6 (*Sub-contribution 2*).

In total, we evaluated 4 real industrial use cases, three from Danobatgroup, and a two networks use case from the Omnifactory case study at the University of Nottingham. To cover time-dependent FBs, we additionally tested an additional use case, which was a simplified FBD from the KNICS project, namely simTRIP.

The feasibility of our testing methodology was evaluated by measuring the success rate of application tests against expected outcomes, which were derived from the test oracles.

The results show that generated test oracles can successfully test IEC 61131-3 AND, and OR logic functions, equality functions from the comparison family, negative trigger edges, and set-reset bistable elements, as well as feedback loops. In contrast, we found that the test oracles could not be used to test TON-timers, given the lack of accuracy of the scan cycles. In addition, TET of the test suite registered more than 1 hour in the case of Omnifactory. This justifies a clear need for a cost-effective selection of test cases, which was addressed in chapter 7.

As a result, this chapter addresses *Hypothesis 3* as:

The results validate that the presented PLC testing methodology can successfully test Siemens PLC programs for non-time dependent FBD programs. However, the methodology was not found to be valid for testing time related FBD programs.

The following table summarises the contributions, objectives and hypotheses accomplished in this chapter:

Table 8.9: Chapter 7 contributions, objectives and hypotheses

Sub-contributions	**Contributions**	**Objectives**	**Hypotheses**
1, 2, 3	Technical Contribution 3	1, 4	3

To the best of our knowledge, this is the first time in which software testing practices have been transferred to Siemens TIA Portal Test Suite to test PLC programs. In future lines, we foresee solving issues related to the scan cycle, thereby addressing a wider range of IEC 61131-3 function groups.

Part V

Final Remarks

Chapter 9

Final Remarks

Contents

This chapter presents the conclusions of this doctoral project. Section 9.1 summarises the contributions, discusses the validation of the hypotheses, and highlights the main limitations of the proposed solutions. Finally, short and mid-term future lines are proposed in Section 9.2.

9.1 Summary of the Contributions

At present, there is unprecedented demand for customised and unique products at low cost. Manufacturing processes need to be dynamically upgraded to meet new customer requirements, meaning that the PLC must manage all necessary changes to implement new bills of processes. As a result, every change should be thoroughly tested to ensure optimal reliability. This is mostly carried out during the commissioning phase, which is the last step of the development process. This issue is further compounded by the fact that commissioning is mostly performed manually, resulting in a time-consuming and error-prone testing process, which delays time to market. Therefore, in this study, we focus on best practices in software engineering, with the aim of transferring this knowledge to industry. The main contributions of this work are as follows:

1. **Preliminary work:** an empirical survey to identify industrial requirements, challenges, and needs (conducted in *Chapter 4*). Industrial and academic stakeholders were consulted, and provided first-hand information on virtual commissioning practices. These results were further reinforced by an expanded review of the literature to determine emerging trends and gaps in the state-of-the-art (explored in *Chapter 3).*

2. **Theoretical framework:** a theoretical virtual commissioning framework designed to address the identified industrial needs (outlined in *Chapter 5*), which ultimately could facilitate the continuous integration of PLCs in manufacturing. The presented theoretical framework addressed the requirements of *Research Focus #1.*

3. **Technical contribution 1:** a test case generation and evaluation approach for IEC 61131-3 FBD programs (described in *Chapter 6*). This approach includes coverage-based, mutation-based, and random test cases. This was achieved by extending the capabilities of existing tools and incorporating a new solver capable of handling a broad range of IEC 61131-3 FBD programs, which includes non-linear arithmetic functions (fulfilling *Gap 7*). An automated test evaluation approach was also developed, in which cost-effective metrics were calculated, and test oracles were obtained from test execution results. The metrics were used to optimise the

selection of the tests in *Chapter 7*, and the test oracles were employed to validate simulation results in *Chapter 8*.

4. **Technical contribution 2:** a cost-effective test selection approach (presented in *Chapter 7*). This involves a search-based test selection algorithm to optimise the selection of test cases. The optimisation focuses on maximising FDC while minimising TET. This was accomplished by defining a fitness function based on the derived cost-effective metrics, which included test case coverage, test case ET, and test case FDC. This contribution covers *Gap 8* and *Gap 10*.

5. **Technical contribution 3:** a methodology to test Siemens PLC programs (detailed in *Chapter 8*). Based on a SiL approach, the solution uses the Siemens automation platform – TIA Portal – to perform tests on the Siemens virtual controller. Test oracles were used to assert the results of the simulation. The methodology comprises four steps, including: 1) PLC code standardisation, 2) test case generation and execution, 3) TIA Portal application test generation, and 4) simulation-based application test execution. This contribution aimed to transfer the existing software engineering practices into industry, tackling *Gap 9*.

Table 9.1 presented below summarises the association of the aforementioned contributions, and the gaps that have been addressed:

Table 9.1: Association between contributions and gaps

Contribution	**Gaps**
Preliminary Work	-
Theoretical framework	3
Technical contribution 1	7
Technical contribution 2	8, 10
Technical contribution 3	9

9.1.1 Hypotheses Validation

We stated four research hypotheses in Section 5.2. This section measures the contributions against these, and determines whether the stated hypotheses have been validated.

Hypothesis 1

The first hypothesis is stated as follows:

"Coverage-based and mutation-based testing enables the automated generation of test cases for most of the IEC 61131-3 function groups, including complex and non-linear arithmetic functions."

This hypothesis is in line with research *Objective 2*. In Chapter 6, we have proposed the Yices 2 SMT2 solver to automatically generate and evaluate test cases for most of the IEC 61131-3 function groups, including non-linear arithmetic, and non-constant division operators. As a result, we extended the capabilities of existing coverage-based and mutation-based testing tools (i.e. FBDTester 2.0, and MuFBDTester) to cover most of the IEC 61131-3 FCs and FBs, as indicated in *Technical Contribution 1*.

We evaluated the proposed approach by employing case studies employed in previous research studies. This comprised 4 different use cases, covering a wide range of IEC 61131-3 function groups, including logic, comparison, arithmetic, timer, counter, bistable, edge trigger, and selection. In addition, we created a new use case to test non-linear arithmetic functions.

The results showed that the new Yices 2 SMT2 solver can generate coverage-based and mutation-based test cases for non-linear arithmetic functions. In addition, we were able to generate test cases for at least the same IEC 61131-3 groups employed in previous studies, which means that the inclusion of the new solver has not limited previous testing capabilities. As a result, we can conclude that the presented solution validates the first hypothesis.

Hypothesis 2

The second hypothesis is stated as follows:

"Search-based test selection algorithms optimise the selection of the generated test cases in a cost-effective manner."

Hypothesis 2 is aligned with *Objective 3*. We have put forward a search-based test selection approach for testing FBD programs, as described in *Chapter 7*. Specifically, we presented a multi-objective test selection approach based on some adequacy criteria, fulfilling *Technical Contribution 2*. To accomplish this, we extended the test generation tools, to include the evaluation of cost-effective metrics calculated in *Technical Contribution 1*. This comprised the calculation of structural coverage measures (i.e. BC, ICC, CCC), FDC and TET. These metrics were then combined to derive 7 fitness functions, which ultimately were used to guide the search algorithm.

The main objective of the presented test selection algorithm was to maximise fault detection while minimising the execution time.

The approach was empirically evaluated with three FBD programs widely used in previous studies. To evaluate the effectiveness of the selected test cases at detecting faults, we employed mutants in the absence of real faults. Specifically, we used two sets of mutants, including second-order mutants. To this end, the experimental scenario consisted of 6 use cases (3 FBD programs x 2 sets of mutants), and it was conducted 50 times given the stochasticity of the search algorithms.

Our results showed that the multi-objective search-based algorithm outperformed the baseline random search in 95.24% of the experiments. We found that the proposed approach significantly reduces time, whilst maintaining a high mutant detection capability, which validates the second hypothesis. In particular, the FDC-based fitness functions were able to cost-effectively detect all mutants. Furthermore, the combination of CCC criterion and TET metric was also found to deliver the best performance, due to its ability to detect faults with only few test cases.

Hypothesis 3

The third hypothesis is stated as follows:

> "The automated PLC testing methodology can test highly configurable industrial PLCs with the use of test oracles."

This research hypothesis corresponds to *Objective 4*, accomplished in *Chapter 8*. We have proposed a simulation-based testing methodology to test the logic of Siemens PLC programs employing test oracles as a basis, in line with *Technical Contribution 3*. We used the Siemens TIA Portal Test Suite to execute the FBD program in the virtual controller PLCSim Advanced. The methodology comprised several steps, including the standardisation of the PLC program, generation of the test cases, execution of the tests to derive test oracles obtained in *Technical Contribution 1*, generation of TIA Test Suite-specific test cases, namely application tests, and lastly simulation-based testing in TIA Portal.

This methodology was evaluated with 5 use cases, four of which were real industrial programs from Danobatgroup and the University of Nottingham. During the experiment, simulation results were asserted with the test oracles and the results were validated by measuring the success rate of the test cases. The experiments were carried out 50 times for those use cases that relied on internal states or values from previous cycles.

The results demonstrate that the generated test oracles can successfully test Siemens PLC programs for non-time dependent FBD programs. The proposed testing methodology can successfully test Siemens PLCs with the use of oracles for the following IEC 61131-3 FCs and FBs: 1) AND and OR logic FCs, 2) equality FCs from comparison family, 3) negative trigger edge FBs, 4) set-reset bistable FBs, and 5) feedback loops. The methodology was not found to be valid for testing timer FBs, however, given the inaccuracy in the cycle duration. Hence, validation of Hypothesis 3 is limited to the aforementioned IEC 61131-3 function FCs and FBs.

9.1.2 Limitations of the Proposed Solutions and the Specific Implementation

This section discusses potential limitations that the proposed solutions might have when applying them in practice.

The FBDTester 3.0 and MuFBDTester 3.0 cover a broad set of IEC 61131-3 function groups, however, these tools are limited mainly to numerical and boolean data type functions. Future research could focus on offering support for the rest of the IEC 61131-3 function groups, including data type conversion functions, bit-shift functions, character strings, functions for time data types, and functions for enumerated data types.

Another point to consider is that the proposed methodology is based on PLCopen XML format IEC 61131-3 projects, hence the methodology promotes interoperability by supporting open standards. However, at present most PLC vendors do not fully follow the IEC 61131 specifications, which limits the ability to exchange PLC programs between different devices. To the best of our knowledge, Codesys Development Systems is one of the few automation software environments that adhere to the PLCopen XML standard. Nowadays, Beckhoff, Schneider Electric, Bosch Rexroth, Toshiba, and Omron claim also to support PLCopen XML standard, which seems that there is a growing interest in this aspect. Despite Siemens being a voting member of PLCopen, the code exported in XML files from the TIA Portal is not compliant with the PLCopen XML template. One possible solution to address this limitation is the development of a conversion tool that parses the information in the exported Siemens XML files into the respective tags indicated in the PLCopen XML standard. Another approach involves exporting Siemens code in lower-level programming languages such as AWL/STL, which partially comply with the IEC 61131-3 IL language. These files could be then processed by Codesys to convert the IL program into the PLCopen XML format. However, Siemens AWL/STL/IL files exhibit some deviations from IEC 61131-3 IL syntax, which require some syntax code modifications, as indicated in

Tables 52-53 of Siemens STEP 7 standards compliance manual [Sie15]. While we expect the industry to make progress in the standardisation and interoperability of PLCs, the need for a conversion tool remains crucial currently.

The test selection algorithm was based on 7 fitness functions, by combining 3 cost-effective metrics, which comprised coverage metrics (i.e. BC, ICC, CCC), FDC, and TET. The fitness objectives derived from these metrics, however, are subject to the objectives established by the stakeholders. The implemented multi-objective search-based algorithm involves a trade-off between time, coverage, and error detection capability.

The proposed methodology to test TIA Portal Siemens PLC programs was restricted to Siemens 1500 PLC families, and required TIA Portal V17 version or newer, as well as PLCs in Advanced V3 version or newer. Siemens is the leader of the European automation market nowadays, in which S7-1500 has emerged as the most popular choice. In the past, the S7-1500 would often be compared to the S7-300 to determine the best solution. Nevertheless, with the obsolescence of S7-300, S7-1500 has become the most popular solution [Arm22], hence this restriction seems a minor issue. In addition, the methodology was constrained to non-time dependent IEC 61131-3 function groups (e.g. TON, TOF, TP). This is a major issue, as timers are used frequently in PLC programs to introduce delays, specify the signal triggering time, and extend signal durations. Consequently, modifications in the main OB network are required to monitor the PLC scan cycle time, and ensure that the output values are asserted in the right scan cycle. Moreover, the approach was only tested with logic and comparison FCs, and bistable, edge detection, and timer FBs. The analysed use cases did not comprise arithmetic (linear/non-linear), numerical and selection FCs, and counter FBs, hence, limiting the scope of applicability. Lastly, is also important to note that the evaluation was limited to two FBD networks, as the validation of *Hypothesis 4* did not prioritise testing additional networks. Nevertheless, adding a second network did not reveal any scalability issues in the proposed testing approach.

Lastly, our approach was tested in an offline virtual controller, rather than on a fully online digital twin. The implementation of the digital twin was beyond the scope of the research project, and was therefore limited to the digital model. One of the potential drawbacks of this is the ability to conduct continuous testing. In the current approach, the PLC code must be retrieved from the real PLC hardware, making a local copy in the simulation environment. Once the new PLC program is successfully tested and commissioned, it must be released in the real PLC. This process requires a high level of version control. In future research, we envisage a fully online solution, in which the PLCSim Advanced is fully synchronised with the real hardware, thus enabling a continuous testing approach.

9.2 Perspectives and Future Work

In this section, we present the short and medium-term objectives to complement this work from three perspectives: industry transfer, application of the proposed methods in a specific domain, adoption of sustainable business models towards innovation, and further research.

9.2.1 Industry Transfer

This study was industry oriented. To this end, we developed a methodology to test Siemens PLC programs, one of the most widely used PLCs. Our approach was evaluated with two industrial case studies, the Omnifactory platform developed at UNOTT, and a machine tool solution used at Danobatgroup, one of the largest European machine tool builders. We envisage transferring this knowledge into industry to further advance in the Technology Readiness Level (TRL) in two ways: 1) showcasing our technology on the Omnifactory platform, which will be used to demonstrate the future assembly platform for the aerospace industry, and 2) working with Danobatgroup to enhance the PLC acceptance process of their machining solutions.

9.2.2 Application of the Proposed Methods in other PLCs

The methods proposed in this dissertation are generic and applicable to other PLC project environments that follow PLCopen XML standards such as Codesys. On another note, despite the fact that *Technical Contribution 3* was limited to Siemens PLC programs, the test oracles could be used to validate the expected behaviour of any PLCs. As a result, other PLC vendors could benefit from adopting this approach when testing their PLC programs. The proposed structural testing methodology could aid other PLC vendors in detecting errors in the PLC program, and thereby, avoid potential delays related to the PLC software in the commissioning process. Although the proposed methodology still requires some manual tasks, overall, all the manual work required for PLC testing might be significantly reduced with the presented approach. However, at the current stage, there are some major barriers to adopting this. Among others, the PLC vendor should support the PLCopen XML standard, or find another alternative to convert it into the standard (as discussed in Section 9.1.2). The proposed methodology has some other limitations, which should be taken into account, such as limitations in the applicability to 2 networks, and to non-time dependent FBs. Besides, TIA Portal execution has not been validated with PLC programs that require arithmetic functions, numerical functions, selection functions, and counters. However, the test oracles could be used to assert the simulation results in other PLC automation software tools, by benchmarking the expected output in the test oracle with the simulated one.

9.2.3 Adoption of Sustainable Business Models towards Innovation

Once appropriate TRL is reached, future work should focus on successfully bringing the presented methodology to the market and sustaining innovation through a spin-off or alliance with a company. This could be achieved by adopting sustainable business models. To this end, we plan to characterise the impact of our approach on sustainable goals established by the European Commission. This characterisation will be carried out by adopting the presented business model in Appendix D towards maximising the reuse and functionality of the manufacturing resources.

9.2.4 Further Research

Further research as well as new developments can be performed to complement this work and we anticipate expanding the empirical evaluations using other IEC 61131-3 function groups, and industrial case studies.

Optimising Processing Time to Generate and Execute Test Cases

Despite optimising the selection of the test cases to reduce execution time, we did not focus on the optimisation of the test case generation time in Java and test case execution in TIA Portal. On the one hand, test case generation time is mainly influenced by the number of mutants, as a Yices 2 SMT2 file needs to be created for each mutant. In addition, the processing time of the Yices 2 SMT2 solver also increases with a high number of coverage requirements. On the other hand, while test optimisation can significantly reduce execution time, particular attention should be paid to other ways of optimising test execution time in TIA Portal. However, this was beyond the scope of this dissertation. Future work should therefore focus on optimising the performance of these tools.

Automating the Conversion of Siemens PLC Programs into PLCopen XML

Given the lack of standardisation of existing commercial solutions available in the market, there is a need to convert vendor-specific PLC programs into PLCopen

XML format. We foresee that the industry will advance on the standardisation and interoperability of PLCs, however a standardisation conversion tool is required at present. As a proof of concept, a manual conversion was enough to evaluate our methodology to test Siemens PLC programs. However, future work should focus on automating this conversion so as to accelerate industry transfer.

Addressing Time-related Issues in TIA Portal

As described in the limitations section, our methodology was constrained to non-time dependent FBD programs. The main issue was the lack of accuracy of the scan cycle time used in the simulation environment, which led to different results each time it was executed. Therefore, the generated test oracles were not able to track the outputs of the theoretical scan cycles accurately. In future lines, we intend to fix this issue by using other mechanisms to measure the runtime of the subject program, or by generating test assertions that handle these issues.

Using a Wider Range of IEC 61131-3 Function Groups

The evaluation of the proposed method was limited to the IEC 61131-3 function groups employed in the use cases. The FBDTester 3.0 and MuFBDTester 3.0 were applicable to most of the function groups, excluding non-numerical and non-boolean data type functions. Thus, FBDTester 3.0 and MuFBDTester 3.0 should include function libraries that support these IEC 61131-3 function groups. In addition, our methodology to test Siemens PLC programs was not evaluated with counters, linear and non-linear arithmetic functions, and selection functions. Future use cases should thus comprise all these functions.

Using a Fully Synchronised Digital Twin to Test Siemens PLC Programs

We have proposed a simulation-based approach to test Siemens PLC programs in a continuous integration environment. At the conclusion of this research project, not all systems were in place, and our case studies were thus conducted in an offline simulation environment, with good results. In future works, however, we envisage implementing a fully online digital twin-based PLC commissioning solution, to enable continuous testing and commissioning, and further support competitiveness in the European manufacturing sector.

Bibliography

[AA16] Ephrem Ryan Alphonsus and Mohammad Omar Abdullah. A review on the applications of programmable logic controllers (PLCs). *Renewable and Sustainable Energy Reviews*, 60:1185–1205, July 2016.

[AB11] Andrea Arcuri and Lionel Briand. A practical guide for using statistical tests to assess randomized algorithms in software engineering. In *Proceedings of the 33rd International Conference on Software Engineering*, ICSE '11, page 1–10, New York, NY, USA, May 2011. ACM.

[ABBZ09] Anne Auger, Johannes Bader, Dimo Brockhoff, and Eckart Zitzler. Theory of the Hypervolume Indicator: Optimal μ-Distributions and the Choice of the Reference Point. In *Proceedings of the Tenth ACM SIGEVO Workshop on Foundations of Genetic Algorithms*, FOGA '09, page 87–102, New York, NY, USA, January 2009. ACM.

[AGN18] Mikel Ayani, Maria Ganebäck, and Amos HC Ng. Digital Twin: Applying emulation for machine reconditioning. In Lihui Wang, editor, *51st CIRP Conference on Manufacturing Systems*, volume 72 of *Procedia CIRP*, pages 243–248. Elsevier, May 2018.

[AGS17] Michael Abramovici, Jens Christian Göbel, and Philipp Savarino. Reconfiguration of smart products during their use phase based on virtual product twins. *CIRP Annals*, 66(1):165–168, April 2017.

[Alu15] Rajeev Alur. *Principles of Cyber-Physical Systems*. The MIT Press, Cambridge, MA, USA, April 2015.

[AMMS18] Raphael Alt, Justus Malzahn, Hubertus Murrenhoff, and Katharina Schmitz. A Survey of Industrial Internet of Things in the Field of Fluid Power: Basic Concept and Requirements for Plug-and-Produce. In *Proceedings of the BATH/ASME 2018 Symposium on Fluid Power and Motion Control*, volume 51968 of *BATH/ASME 2018*, page 11, New York, NY, USA, November 2018. American Society of Mechanical Engineers.

[AP12] Shivani Acharya and Vidhi Pandya. Bridge between black box and white box–gray box testing technique. *International Journal of Electronics and Computer Science Engineering*, 2(1):175–185, 2012.

[Arm22] Antonio Armenta. Siemens SIMATIC PLCs - Hardware History. https://control.com/technical-articles/siemens-simatic-plcs-hardware-history/, May 2022. Technical article.

[Arr22] Aitor Arrieta. Is the Revisited Hypervolume an Appropriate Quality Indicator to Evaluate Multi-Objective Test Case Selection Algorithms? In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO ’22, page 1317–1326, New York, NY, USA, July 2022. ACM.

[ASEZ17] Aitor Arrieta, Goiuria Sagardui, Leire Etxeberria, and Justyna Zander. Automatic Generation of Test System Instances for Configurable Cyber-Physical Systems. *Software Quality Journal*, 25(3):1041–1083, September 2017.

[AVAS23] Aitor Arrieta, Pablo Valle, Joseba A. Agirre, and Goiuria Sagardui. Some Seeds Are Strong: Seeding Strategies for Search-Based Test Case Selection. *ACM Transactions on Software Engineering and Methodology*, 32(1), February 2023.

[AVB99] Franz Auinger, Markus Vorderwinkler, and Georg Buchtela. Interface Driven Domain-Independent Modeling Architecture for “Soft-Commissioning” and “Reality in the Loop”. In *Proceedings of the 31st Conference on Winter Simulation: Simulation—a Bridge to the Future*, volume 1 of *WSC ’99*, page 798–805, New York, NY, USA, December 1999. ACM.

[AWA$^+$18] Aitor Arrieta, Shuai Wang, Ainhoa Arruabarrena, Urtzi Markiegi, Goiuria Sagardui, and Leire Etxeberria. Multi-objective black-box test case selection for cost-effectively testing simulation models. In

Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '18, page 1411–1418, New York, NY, USA, July 2018. ACM.

[AWM+19] Aitor Arrieta, Shuai Wang, Urtzi Markiegi, Ainhoa Arruabarrena, Leire Etxeberria, and Goiuria Sagardui. Pareto efficient multi-objective black-box test case selection for simulation-based testing. *Information and Software Technology*, 114:137–154, October 2019.

[AWSE19] Aitor Arrieta, Shuai Wang, Goiuria Sagardui, and Leire Etxeberria. Search-based test case prioritization for simulation-based testing of cyber-physical system product lines. *Journal of Systems and Software*, 149:1–34, 2019.

[Bat96] Robert N Bateson. *Introduction to Control System Technology*. Printice Hall, 1996.

[BCMV17] E. Bottani, A. Cammardella, T. Murino, and S. Vespoli. From the cyber-physical system to the digital twin: The process development for behaviour modelling of a cyber guided vehicle in M2M logic. In *Proceedings of the Summer School Francesco Turco*, 22nd Summer School "Francesco Turco" - Industrial Systems Engineering, page 96 – 102, September 2017.

[BDL04] Gerd Behrmann, Alexandre David, and Kim G. Larsen. A Tutorial on UPPAAL. In Marco Bernardo and Flavio Corradini, editors, *Formal Methods for the Design of Real-Time Systems: International School on Formal Methods for the Design of Computer, Communication, and Software Systems*, volume 3185 of *Lecture Notes in Computer Science*, pages 200–236. Springer, Berlin, Heidelberg, 2004.

[Bek05] George A Bekey. *Autonomous robots: from biological inspiration to implementation and control*. The MIT press, Cambridge, MA, USA, February 2005.

[BG11] Radhakisan Baheti and Helen Gill. Cyber-physical systems. The impact of control technology. *Open Journal of Social Sciences*, 12(1):161–166, November 2011.

[BGS+18] Ron Bitton, Tomer Gluck, Orly Stan, Masaki Inokuchi, Yoshinobu Ohta, Yoshiyuki Yamada, Tomohiko Yagyu, Yuval Elovici, and Asaf

Shabtai. Deriving a cost-effective digital twin of an ICS to facilitate security evaluation. In *Computer Security*, volume 11098, pages 533–554, Cham, August 2018. Springer.

[BHO$^+$18] Darya Botkina, Mikael Hedlind, Bengt Olsson, Jannik Henser, and Thomas Lundholm. Digital Twin of a Cutting Tool. In Lihui Wang, editor, *51st CIRP Conference on Manufacturing Systems*, volume 72 of *Procedia CIRP*, pages 215–218. Elsevier, May 2018.

[BMKW18] Florian Biesinger, Davis Meike, Benedikt Kraß, and Michael Weyrich. A digital twin for production planning based on cyber-physical systems: A Case Study for a Cyber-Physical System-Based Creation of a Digital Twin. In Roberto Teti and Doriana M. D'Addona, editors, *12th CIRP Conference on Intelligent Computation in Manufacturing Engineering*, volume 79 of *Procedia CIRP*, pages 355–360. Elsevier, July 2018.

[BMMP00] Luciano Baresi, Marco Mauri, Antonello Monti, and Mauro Pezze. PLCTools: design, formal validation, and code generation for programmable controllers. In *IEEE International Conference on Systems, Man Cybernetics.'Cybernetics evolving to systems, humans, organizations, and their complex interactions'*, volume 4 of *SMC 2000 Conference Proceedings*, pages 2437–2442. IEEE, October 2000.

[BS19] Logica Banica and Cristian Stefan. Stepping into the Industry 4.0: The Digital Twin Approach. *Annals of the University Dunarea de Jos of Galati: Fascicle: I, Economics & Applied Informatics*, 25(3):107–113, 2019.

[BSF$^+$16] Dimitri Bohlender, Hendrik Simon, Nico Friedrich, Stefan Kowalewski, and Stefan Hauck-Stattelmann. Concolic test generation for PLC programs using coverage metrics. In *13th International Workshop on Discrete Event Systems*, WODES, pages 432–437. IEEE, June 2016.

[BVBS19] Enrique Blanco Viñuela, João Borrego, and B Schofield. Continuous Integration for PLC-Based Control Systems. In *17th International Conference on Accelerator and Large Experimental Physics Control Systems*, ICALEPCS 2019. JACoW, October 2019.

[CMK+22] Anthony Corso, Robert J. Moss, Mark Koren, Ritchie Lee, and Mykel J. Kochenderfer. A survey of algorithms for black-box safety validation of cyber-physical systems. *Journal of Artificial Intelligence Research*, 72:377–428, January 2022.

[CSCL17] Yi Cai, Binil Starly, Paul Cohen, and Yuan-Shin Lee. Sensor Data and Information Fusion to Construct Digital-twins Virtual Machine Tools for Cyber-physical Manufacturing. In Lihui Wang, Livan Fratini, and Albert J. Shih, editors, *45th SME North American Manufacturing Research Conference*, volume 10 of *Procedia Manufacturing*, pages 1031–1042. Elsevier, June 2017.

[CTC17] Jian-Yu Chen, Kuo-Cheng Tai, and Guo-Chin Chen. Application of Programmable Logic Controller to Build-up an Intelligent Industry 4.0 Platform. In Mitchell M. Tseng, Hung-Yin Tsai, and Yue Wang, editors, *50th CIRP Conference on Manufacturing Systems*, volume 63 of *Procedia CIRP*, pages 150–155. Elsevier, May 2017.

[CW20] Benoit Combemale and Manuel Wimmer. Towards a model-based devops for cyber-physical systems. In Bertrand Meyer Jean-Michel Bruel, Manuel Mazzara, editor, *Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment*, volume 12055 of *Lecture Notes in Computer Science*, pages 84–94, Cham, January 2020. Springer.

[DB18] Violeta Damjanovic-Behrendt. A Digital Twin-based Privacy Enhancement Mechanism for the Automotive Industry. In *9th International Conference on Intelligent Systems*, IS 2018, pages 272–279. IEEE, September 2018.

[DCZ+19] Kai Ding, Felix TS Chan, Xudong Zhang, Guanghui Zhou, and Fuqiang Zhang. Defining a Digital Twin-based Cyber-Physical Production System for autonomous manufacturing in smart shop floors. *International Journal of Production Research*, 57(20):6315–6334, October 2019.

[DDG+18] Lorenzo Damiani, Melissa Demartini, Piero Giribone, Margherita Maggiani, Roberto Revetria, and Flavio Tonelli. Simulation and digital twin based design of a production line: a case study. In Oscar Castillo, David Dagan Feng, A.M. Korsunsky, Craig Douglas, and

S.I. Ao, editors, *2018 International MultiConference of Engineers and Computer Scientists*, volume 2 of *Lecture Notes in Engineering and Computer Science*. Newswood Limited, International Association of Engineers, IAENG, March 2018.

[DDM06] Bruno Dutertre and Leonardo De Moura. *The Yices SMT Solver*. Computer Science Laboratory, SRI International, Menlo Park, CA, USA, 2006.

[DE$^+$18] Alexander Detzner, Martin Eigner, et al. A digital twin for root cause analysis and product quality monitoring. In *15th International Design Conference*, volume 4, pages 1547–1558. The Design Society, May 2018.

[DEA$^+$19] Davide D'Amico, John Ekoyuncu, Sri Addepalli, Christopher Smith, Ed Keedwell, Jim Sibson, and Steven Penver. Conceptual framework of a digital twin to evaluate the degradation status of complex engineering systems. In Franz Dietrich and Nicole Krenkel, editors, *7th CIRP Global Web Conference*, volume 86 of *Procedia CIRP*, pages 61–67. Elsevier, October 2019.

[DJK12] Rupesh Dev, Antti Jääskeläinen, and Mika Katara. Model-Based GUI Testing. Case Smartphone Camera and Messaging Development. *Advances in Computers*, 85:65–122, April 2012.

[Do16] Hyunsook Do. Recent advances in regression testing techniques. *Advances in computers*, 103:53–77, May 2016.

[DPAM02] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.

[DR18a] Ulrich Dahmen and Juergen Rossmann. Experimentable Digital Twins for a Modeling and Simulation-based Engineering Approach. In *2018 IEEE International Systems Engineering Symposium*, ISSE, pages 1–8. IEEE, November 2018.

[DR18b] Ulrich Dahmen and Jürgen Roßmann. Simulation-based Verification with Experimentable Digital Twins in Virtual Testbeds. In Thorsten Schüppstuhl, Kirsten Tracht, and Jörg Franke, editors, *Tagungsband* *des 3. Kongresses Montage Handhabung Industrieroboter*, pages 139–147. Springer Vieweg, Berlin, Heidelberg, April 2018.

[dSdABG+08] Leandro Dias da Silva, Luiz Paulo de Assis Barbosa, Kyller Gorgonio, Angelo Perkusich, and Antonio Marcus Nogueira Lima. On the automatic generation of timed automata models from Function Block Diagrams for safety instrumented systems. In *34th Annual Conference of IEEE Industrial Electronics*, pages 291–296. IEEE, November 2008.

[Dut14] Bruno Dutertre. *Yices 2 manual.* Computer Science Laboratory, SRI International, Menlo Park, CA, USA, 2014.

[EA20] Montero Vera Edgar Alexander. Virtual commissioning of an industrialwood cutter machine : A software in the loop simulation. Technical report, Luleå University of Technology, Department of Computer Science, Electrical and Space Engineering, Luleå, Sweden, January 2020. Degree Project.

[EČO+16] Eduard P Enoiu, Adnan Čaušević, Thomas J Ostrand, Elaine J Weyuker, Daniel Sundmark, and Paul Pettersson. Automated Test Generation Using Model Checking: An Industrial Evaluation. *International Journal on Software Tools for Technology Transfer*, 18(3):335–353, June 2016.

[EDB+13] Eduard Paul Enoiu, Kivanc Doganay, Markus Bohlin, Daniel Sundmark, and Paul Pettersson. MOS: An Integrated Model-Based and Search-Based Testing Tool for Function Block Diagrams. In *Proceedings of the 1st International Workshop on Combining Modelling and Search-Based Software Engineering*, CMSBSE '13, page 55–60. IEEE Press, May 2013.

[EE18] Matthias Eckhart and Andreas Ekelhart. Towards security-aware virtual environments for digital twins. In *Proceedings of the 4th ACM Workshop on Cyber-Physical System Security*, CPSS '18, page 61–72, New York, NY, USA, May 2018. ACM.

[EGHS16] Christof Ebert, Gorka Gallardo, Josune Hernantes, and Nicolas Serrano. DevOps. *IEEE Software*, 33(3):94–100, 2016.

[EKG90] W Eversheim, D Koerth, and J Gentzcke. Gesellschaft produktionstechnik: Inbetriebnahme komplexer maschinen und anlagen:

Strategien und praxisbeispiele zur rationalisierung in der einzel-und kleinserienproduktion. *Düsseldorf: VDI-Verl*, 1990.

[EMLH10] E Estévez, Marga Marcos, Arndt Lüder, and Lorenz Hundt. PLCopen for achieving interoperability between development phases. In *IEEE 15th Conference on Emerging Technologies & Factory Automation*, ETFA 2010, pages 1–8. IEEE, September 2010.

[ERS10] Emelie Engström, Per Runeson, and Mats Skoglund. A systematic review on regression test selection techniques. *Information and Software Technology*, 52(1):14–30, January 2010.

[ESP13] Eduard Paul Enoiu, Daniel Sundmark, and Paul Pettersson. Model-Based Test Suite Generation for Function Block Diagrams Using the UPPAAL Model Checker. In *2013 IEEE sixth international conference on software testing, verification and validation workshops*, pages 158–167. IEEE, March 2013.

[ESv$^+$16] Eduard P. Enoiu, Daniel Sundmark, Adnan Čaušević, Robert Feldt, and Paul Pettersson. Mutation-Based Test Generation for PLC Embedded Software Using Model Checking. In Franz Wotawa, Mihai Nica, and Natalia Kushik, editors, *Testing Software and Systems*, volume 9976 of *Lecture Notes in Computer Science*, page 155–171, Cham, October 2016. Springer.

[FBM13] B Fernández, E Blanco, and A Merezhin. Testing & verification of PLC code for process control. In *17th International Conference on Accelerator and Large Experimental Physics Control Systems*, ICALEPCS'13, pages 1258–1261, 2013.

[GAA$^+$21] Aitor Gartziandia, Jon Ayerdi, Aitor Arrieta, Shaukat Ali, Tao Yue, Aitor Agirre, Goiuria Sagardui, and Maite Arratibel. Microservices for Continuous Deployment, Monitoring and Validation in Cyber-Physical Systems: an Industrial Case Study for Elevators Systems. In *2021 IEEE 18th International Conference on Software Architecture Companion*, ICSA-C, pages 46–53. IEEE, March 2021.

[Gar17] Boni Garcia. *Mastering Software Testing with JUnit 5: Comprehensive guide to develop high quality Java applications.* Packt Publishing, October 2017.

[GBK+16] Thomas Gabor, Lenz Belzner, Marie Kiermeier, Michael Till Beck, and Alexander Neitz. A simulation-based architecture for smart cyber-physical systems. In *2016 IEEE international conference on autonomic computing*, pages 374–379. IEEE, July 2016.

[GC19] Jokin Garcia and Jordi Cabot. Stepwise adoption of continuous delivery in model-driven engineering. In Bertrand Meyer Jean-Michel Bruel, Manuel Mazzara, editor, *Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment*, volume 11350 of *Lecture Notes in Computer Science*, pages 19–32, Cham, March 2019. Springer.

[GGWB17] Edward R Griffor, Christopher Greer, David A Wollman, and Martin J Burns. Framework for cyber-physical systems: Volume 1, overview. Technical report, NIST Special Publication 1500-201, June 2017. Version 1.0.

[Gil06] Helen Gill. *NSF Perspective and Status on Cyber-physical Systems*. National Workshop on Cyber-physical Systems. National Science Foundation, Austin, TX, USA, October 2006.

[Gri05] Michael W Grieves. Product lifecycle management: the new paradigm for enterprises. *International Journal of Product Development*, 2(1-2):71–84, January 2005.

[GŚ12] Arkadiusz Gola and Antoni Świć. Directions of Manufacturing systems' evolution from the flexibility level point of view. *Innovations in Management and Production Engineering*, pages 226–238, January 2012.

[GSWH15] Gregory Gay, Matt Staats, Michael Whalen, and Mats PE Heimdahl. The risks of coverage-directed test case generation. *IEEE Transactions on Software Engineering*, 41(8):803–819, August 2015.

[GZSZ19] Jiapeng Guo, Ning Zhao, Lin Sun, and Saipeng Zhang. Modular based flexible digital twin for factory design. *Journal of Ambient Intelligence and Humanized Computing*, 10(3):1189–1200, March 2019.

[Har11] Mark Harman. Making the case for MORTO: Multi objective regression test optimization. In *IEEE Fourth International Conference* *on Software Testing, Verification and Validation Workshops*, pages 111–114. IEEE, March 2011.

[HB10] Hadi Hemmati and Lionel Briand. An industrial investigation of similarity measures for model-based test case selection. In *IEEE 21st International Symposium on Software Reliability Engineering*, pages 141–150. IEEE, November 2010.

[HF06] Tanvir Hussain and Georg Frey. UML-based development process for IEC 61499 with automatic test-case generation. In *IEEE Conference on Emerging Technologies and Factory Automation*, pages 1277–1284. IEEE, September 2006.

[HG$^+$17] Dominik Heber, Marco Groll, et al. Towards a digital twin: How the blockchain can foster E/E-traceability in consideration of model-based systems engineering. In Anja Maier, Stanko Škec, Harrison Kim, Michael Kokkolaras, Josef Oehmen, Georges Fadel, Filippo Salustri, and Mike Van der Loos, editors, *1st International Conference on Engineering Design*, volume 3 of *Product, Services and Systems Design*, pages 321–330. The Design Society, August 2017.

[HHHY20] Jerome Hugues, Anton Hristosov, John J. Hudak, and Joe Yankel. Twinops - devops meets model-based engineering and digital twins for the engineering of cps. In *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, MODELS '20, New York, NY, USA, October 2020. ACM.

[HHL$^+$19] Wilhelm Hasselbring, Sören Henning, Björn Latte, Armin Möbius, Thomas Richter, Stefani Schalk, and Maik Wojcieszak. Industrial DevOps. In *2019 IEEE International Conference on Software Architecture Companion*, ICSA-C, pages 123–126. IEEE, March 2019.

[HKVH$^+$11] Reinhard Hametner, Benjamin Kormann, Birgit Vogel-Heuser, Dietmar Winkler, and Alois Zoitl. Test case generation approach for industrial automation systems. In *The 5th International Conference on Automation, Robotics and Applications*, pages 57–62. IEEE, December 2011.

[HKVH$^+$13] Reinhard Hametner, Benjamin Kormann, Birgit Vogel-Heuser, Dietmar Winkler, and Alois Zoitl. *Recent Advances in Robotics and*

Automation, volume 480 of *Studies in Computational Intelligence*, chapter Automated Test Case Generation for Industrial Control Applications, pages 263–273. Springer, Berlin, Heidelberg, 2013.

[HSMP10] Peter Hoffmann, Reimar Schumann, Talal MA Maksoud, and Giuliano C Premier. Virtual Commissioning Of Manufacturing Systems A Review And New Approaches For Simplification. In *Proceedings of the 24th European Conference on Modelling and Simulation*, ECMS 2010, pages 175–181. CAL-TEK SRL, June 2010.

[HUL$^+$18] Wladimir Hofmann, Jan Hendrik Ulrich, Sebastian Lang, Tobias Reggelin, and Juri Tolujew. Simulation and Virtual Commissioning of Modules for a Plug-and-Play Conveying System. In *16th IFAC Symposium on Information Control Problems in Manufacturing*, volume 51 of *IFAC-PapersOnLine*, pages 649–654. Elsevier, June 2018.

[HVF] Csaba Hegedũs, Pál Varga, and Attila Frankó. A DevOps Approach for Cyber-Physical System-of-Systems Engineering through Arrowhead. In *2021 IFIP/IEEE International Symposium on Integrated Network Management*, IM, May.

[IH14] Laura Inozemtseva and Reid Holmes. Coverage is not strongly correlated with test suite effectiveness. In *Proceedings of the 36th International Conference on Software Engineering*, ICSE 2014, page 435–445, New York, NY, USA, May 2014. ACM.

[ind06] Software Design Specification for the Bistable Processor of the Reactor Protection System. Technical report, Doosan Heavy Industry and Construction, 2006. KNICS-RPS-SDS231-01, Rev. 01.

[ISS99] Rolf Isermann, Jochen Schaffnit, and Stefan Sinsel. Hardware-in-the-loop simulation for the design and testing of engine-control systems. *Control Engineering Practice*, 7(5):643–653, May 1999.

[Jan18] Petr Janda. Mechatronic concept of heavy machine tools. In Branko Katalinic, editor, *Proceedings of the 29th International DAAAM Symposium 2018*, volume 29, Vienna, Austria, January 2018. DAAAM International.

[JH10] Yue Jia and Mark Harman. An analysis and survey of the development of mutation testing. *IEEE transactions on software engineering*, 37(5):649–678, June 2010.

[JJC+10] Eunkyoung Jee, Seungjae Jeon, Sungdeok Cha, Kwangyong Koh, Junbeom Yoo, Geeyong Park, and Poonghyun Seong. FBDVerifier: Interactive and Visual Analysis of Counter-example in Formal Verification of Function Block Diagram. *Journal of Research and Practice in Information Technology*, 42(3):171–188, September 2010.

[JJI+14] René Just, Darioush Jalali, Laura Inozemtseva, Michael D. Ernst, Reid Holmes, and Gordon Fraser. Are Mutants a Valid Substitute for Real Faults in Software Testing? In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, FSE 2014, page 654–665, New York, NY, USA, November 2014. ACM.

[Jov17] Dejan Jovanović. Solving nonlinear integer arithmetic with MCSAT. In Ahmed Bouajjani and David Monniaux, editors, *International Conference on Verification, Model Checking, and Abstract Interpretation*, volume 10145 of *Lecture Notes in Computer Science*, pages 330–346, Cham, January 2017. Springer.

[JSB18] E Jee, J Song, and DH Bae. Definition and application of mutation operator extensions for FBD programs. *KIISE Transactions on Computing Practices*, 24(11):589–595, November 2018.

[JSC+14] Eunkyoung Jee, Donghwan Shin, Sungdeok Cha, Jang-Soo Lee, and Doo-Hwan Bae. Automated test case generation for FBD programs implementing reactor protection system software. *Software Testing, Verification and Reliability*, 24(8):608–628, September 2014.

[JSN+20] David Jones, Chris Snider, Aydin Nassehi, Jason Yon, and Ben Hicks. Characterising the Digital Twin: A systematic literature review. *CIRP Journal of Manufacturing Science and Technology*, 29:36–52, May 2020.

[JYC05] Eunkyoung Jee, Junbeom Yoo, and Sungdeok Cha. Control and Data Flow Testing on Function Block Diagrams. In Winther Rune, Axel Gran Bjørn, and Dahll Gustav, editors, *International Conference on Computer Safety, Reliability, and Security*, volume 3688 of

Lecture Notes in Computer Science, pages 67–80, Berlin, Heidelberg, September 2005. Springer.

[JYCB09] Eunkyoung Jee, Junbeom Yoo, Sungdeok Cha, and Doohwan Bae. A data flow-based structural testing technique for FBD programs. *Information and Software Technology*, 51(7):1131–1139, July 2009.

[KCK19] Sungjoo Kang, Ingeol Chun, and Hyeon-Soo Kim. Design and Implementation of Runtime Verification Framework for Cyber-Physical Production Systems. *Journal of Engineering*, 2019, November 2019.

[KGK$^+$19] J Koziorek, A Gavlas, J Konecny, M Mikolajek, R Kraut, and P Walder. Automated control system design with model-based commissioning. *International Journal of Circuits, Systems and Signal Processing*, 13(2019):6–12, 2019.

[KHJ$^+$99] Yoram Koren, Uwe Heisel, Francesco Jovane, Toshimichi Moriwaki, Gumter Pritschow, Galip Ulsoy, and Hendrik Van Brussel. Reconfigurable manufacturing systems. *CIRP Annals*, 48(2):527–540, January 1999.

[KK$^+$12] Mohd Ehmer Khan, Farmeena Khan, et al. A comparative study of white box, black box and grey box testing techniques. *International Journal of Advanced Computer Science and Applications*, 3(6), June 2012.

[KKT$^+$18] Werner Kritzinger, Matthias Karner, Georg Traar, Jan Henjes, and Wilfried Sihn. Digital Twin in manufacturing: A categorical literature review and classification. In *16th IFAC Symposium on Information Control Problems in Manufacturing*, volume 51 of *IFAC-PapersOnLine*, pages 1016–1022. Elsevier, July 2018.

[KMT$^+$17] Vladimir Kuts, Gianfranco E Modoni, Walter Terkaj, Toivo Tähemaa, Marco Sacco, and Tauno Otto. Exploiting Factory Telemetry to Support Virtual Reality Simulation in Robotics Cell. In Lucio Tommaso De Paolis, Patrick Bourdot, and Antonio Mongelli, editors, *Augmented Reality, Virtual Reality, and Computer Graphics*, volume 10324 of *Lecture Notes in Computer Science*, pages 212–221, Cham, June 2017. Springer.

[KPL$^+$11] Lock-Jo Koo, Chang Mok Park, Chang Ho Lee, SangChul Park, and Gi-Nam Wang. Simulation framework for the verification of

PLC programs in automobile industries. *International Journal of Production Research*, 49(16):4925–4943, August 2011.

[KWL+20] Achim Kampker, Saskia Wessel, Nicolas Lutz, Michaela Reibetanz, and Martin Hehl. Virtual Commissioning for Scalable Production Systems in the Automotive Industry: Model for evaluating benefit and effort of virtual commissioning. In *9th International Conference on Industrial Technology and Management*, ICITM 2020, pages 107–111. IEEE, February 2020.

[Lah14] Jussi Lahtinen. Automatic Test Set Generation for Function Block Based Systems Using Model Checking. In *9th International Conference on the Quality of Information and Communications Technology*, pages 216–225. IEEE, Semptember 2014.

[LFM+19] Tobias Lechler, Eva Fischer, Maximilian Metzner, Andreas Mayr, and Jörg Franke. Virtual Commissioning–Scientific review and exploratory use cases in advanced production systems. In Butala Peter, Govekar Edvard, and Vrabič Rok, editors, *52nd CIRP Conference on Manufacturing Systems*, volume 81 of *Procedia CIRP*, pages 1125–1130. Elsevier, June 2019.

[LFN+17] Remo Lachmann, Michael Felderer, Manuel Nieke, Sandro Schulze, Christoph Seidl, and Ina Schaefer. Multi-Objective Black-Box Test Case Selection for System Testing. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '17, page 1311–1318, New York, NY, USA, July 2017. ACM.

[LJB22a] Lingjun Liu, Eunkyoung Jee, and Doo-Hwan Bae. Mufbdtester: A mutation-based test sequence generator for fbd programs implementing nuclear power plant software. *Software Testing, Verification and Reliability*, page e1815, 2022.

[LJB22b] Lingjun Liu, Eunkyoung Jee, and Doo-Hwan Bae. MuFBDTester: A mutation-based test sequence generator for FBD programs implementing nuclear power plant software. *Software Testing, Verification and Reliability*, 32(8):e1815, May 2022.

[LKY21] Dong-Ah Lee, Eui-Sub Kim, and Junbeom Yoo. Quantitative measures of thoroughness of FBD simulations for PLC-based digital

I&C system. *Nuclear Engineering and Technology*, 53(1):131–141, January 2021.

[LLKM18] Minna Lanz, Andrei Lobov, Kati Katajisto, and Petteri Mäkelä. A concept and local implementation for industry-academy collaboration and life-long learning. In Mourtzis Dimitris and Chryssolouris George, editors, *Advanced Engineering Education Training for Manufacturing Innovation, 8th CIRP Sponsored Conference on Learning Factories*, volume 23 of *Procedia Manufacturing*, pages 189–194. Elsevier, April 2018.

[LLTY15] Bingdong Li, Jinlong Li, Ke Tang, and Xin Yao. Many-Objective Evolutionary Algorithms: A Survey. *ACM Computing Surveys*, 48(1):1–35, September 2015.

[LLY$^+$20] Jiewu Leng, Qiang Liu, Shide Ye, Jianbo Jing, Yan Wang, Chaoyang Zhang, Ding Zhang, and Xin Chen. Digital twin-driven rapid reconfiguration of the automated manufacturing system via an open architecture model. *Robotics and Computer-Integrated Manufacturing*, 63(C):101895, June 2020.

[LM18] Francesca Lonetti and Eda Marchetti. Emerging software testing technologies. In Atif M. Memon, editor, *Advances in Computers*, volume 108, pages 91–143. Elsevier, 2018.

[LML$^+$17] Chenzhao Li, Sankaran Mahadevan, You Ling, Sergio Choze, and Liping Wang. Dynamic Bayesian Network for Aircraft Wing Health Monitoring Digital Twin. *AIAA Journal*, 55(3):930–941, January 2017.

[LP05] Abdesselam Lakehal and Ioannis Parissis. Structural Test Coverage Criteria for Lustre Programs. In *Proceedings of the 10th International Workshop on Formal Methods for Industrial Critical Systems*, FMICS '05, page 35–43, New York, NY, USA, September 2005. ACM.

[LP14] Chi G Lee and Sang C Park. Survey on the virtual commissioning of manufacturing systems. *Journal of Computational Design and Engineering*, 1(3):213–222, July 2014.

[LRP16] Reinhard Langmann and Leandro F Rojas-Peña. A PLC as an Industry 4.0 component. In *13th International Conference on Remote*

Engineering and Virtual Instrumentation, REV 2016, pages 10–15. IEEE, February 2016.

[LW89] Hareton KN Leung and Lee White. Insights into regression testing (software testing). In *Proceedings. Conference on Software Maintenance*, pages 60–69. IEEE, October 1989.

[LWIL20] Yiling Lai, Yuchen Wang, Robert Ireland, and Ang Liu. Digital twin driven virtual verification. In *Digital Twin Driven Smart Design*, pages 109–138. Elsevier, 2020.

[LZLC19] Qiang Liu, Hao Zhang, Jiewu Leng, and Xin Chen. Digital twin-driven rapid individualised designing of automated flow-shop manufacturing system. *International Journal of Production Research*, 57(12):3903–3919, May 2019.

[LZY$^+$19] Jiewu Leng, Hao Zhang, Douxi Yan, Qiang Liu, Xin Chen, and Ding Zhang. Digital twin-driven manufacturing cyber-physical system for parallel controlling of smart workshop. *Journal of Ambient Intelligence and Humanized Computing*, 10(3):1155–1166, June 2019.

[Mac14] William MacDougall. *Industrie 4.0 Smart Manufacturing for the future*. Future markets. Germany Trade & Invest, July 2014.

[Mad00] Angelika Mader. *Discrete Event Systems: Analysis and Control*, volume 569 of *The Springer International Series in Engineering and Computer Science*, chapter A Classification of PLC Models and Applications, pages 239–246. Springer, Boston, MA, USA, May 2000.

[MAH18] Alexander McDermott Miller, Ramon Alvarez, and Nathan Hartman. Towards an extended model-based definition for the digital twin. *Computer-Aided Design and Applications*, 15(6):880–891, April 2018.

[MASE17] Urtzi Markiegi, Aitor Arrieta, Goiuria Sagardui, and Leire Etxeberria. Search-Based Product Line Fault Detection Allocating Test Cases Iteratively. In *Proceedings of the 21st International Systems and Software Product Line Conference*, volume A of *SPLC '17*, page 123–132, New York, NY, USA, September 2017. ACM.

[MCS$^+$22] Fan Mo, Jack C. Chaplin, David Sanderson, Hamood Ur Rehman, Fabio Marco Monetti, Antonio Maffei, and Svetan Ratchev. A Framework for Manufacturing System Reconfiguration Based on Artificial Intelligence and Digital Twin. In Kyoung-Yun Kim, Leslie Monplaisir, and Jeremy Rickli, editors, *Flexible Automation and Intelligent Manufacturing: The Human-Data-Technology Nexus*, Lecture Notes in Mechanical Engineering, pages 361–373, Cham, June 2022. Springer.

[MGN19] Antonio Maffei, Sten Grahn, and Cali Nuur. Characterization of the impact of digitalization on the adoption of sustainable business models in manufacturing. In Butala Peter, Govekar Edvard, and Vrabič Rok, editors, *52nd CIRP Conference on Manufacturing Systems*, volume 81, pages 765–770. Elsevier, June 2019.

[MKB$^+$16] László Monostori, Botond Kádár, Thomas Bauernhansl, Shinsuke Kondoh, S Kumara, Gunther Reinhart, Olaf Sauer, Gunther Schuh, Wilfried Sihn, and Kenichi Ueda. Cyber-physical systems in manufacturing. *CIRP Annals*, 65(2):621–641, August 2016.

[MN10] Patrick E McKight and Julius Najab. Kruskal-Wallis Test. *The Corsini Encyclopedia of Psychology*, pages 1–1, January 2010.

[MPPU19] Claudio Mandolla, Antonio Messeni Petruzzelli, Gianluca Percoco, and Andrea Urbinati. Building a digital twin for additive manufacturing through the exploitation of blockchain: A case analysis of the aircraft industry. *Computers in Industry*, 109:134–152, April 2019.

[MPZ15] Leonardo Mariani, Mauro Pezze, and Daniele Zuddas. Recent advances in automatic black-box testing. In Atif Memon, editor, *Advances in computers*, volume 99, pages 157–193. Elsevier, December 2015.

[MRNF18] Marco Macchi, Irene Roda, Elisa Negri, and Luca Fumagalli. Exploring the role of digital twin for asset lifecycle management. In Marco Macchi, László Monostori, and Roberto Pinto, editors, *16th IFAC Symposium on Information Control Problems in Manufacturing*, volume 51 of *IFAC-PapersOnLine*, pages 790–795. Elsevier, June 2018.

[MSS14] Kohei Maruchi, Hiromasa Shin, and Masahiro Sakai. MC/DC-like structural coverage criteria for function block diagrams. In *IEEE Seventh International Conference on Software Testing, Verification and Validation Workshops*, pages 253–259. IEEE, June 2014.

[ND12] Srinivas Nidhra and Jagruthi Dondeti. Black Box and White Box Testing Techniques – A Literature Review. *International Journal of Embedded Systems and Applications*, 2(2):29–50, June 2012.

[NPHS16] Muhammad Nouman, Usman Pervez, Osman Hasan, and Kashif Saghar. Software testing: A survey and tutorial on white and black-box testing of C/C++ programs. In *IEEE Region 10 Symposium*, TENSYMP 2016, pages 225–230. IEEE, July 2016.

[NT11] Kshirasagar Naik and Priyadarshi Tripathy. *Software Testing and Quality Assurance: Theory and Practice*. John Wiley & Sons, September 2011.

[PDK19] Diego Galar Pascual, Pasquale Daponte, and Uday Kumar. *Handbook of Industry 4.0 and SMART Systems*. CRC Press, September 2019.

[PE10] Olivera Pavlovic and Hans-Dieter Ehrich. Model checking PLC software written in function block diagram. In *Third International Conference on Software Testing, Verification and Validation*, pages 439–448. IEEE, June 2010.

[PKT17] Annibale Panichella, Fitsum Meshesha Kifetew, and Paolo Tonella. Automated Test Case Generation as a Many-Objective Optimisation Problem with Dynamic Selection of the Targets. *IEEE Transactions on Software Engineering*, 44(2):122–158, February 2017.

[PKZ$^+$19] Mike Papadakis, Marinos Kintis, Jie Zhang, Yue Jia, Yves Le Traon, and Mark Harman. Mutation Testing Advances: An Analysis and Survey. In Atif M. Memon, editor, *Advances in Computers*, volume 112, pages 275–378. Elsevier, 2019.

[PMLK13] Antti Pakonen, Teemu Mätäsniemi, Jussi Lahtinen, and Tommi Karhela. A toolset for model checking of PLC software. In *IEEE 18th Conference on Emerging Technologies & Factory Automation*, ETFA 2013, pages 1–6. IEEE, October 2013.

[PODPDL14] Annibale Panichella, Rocco Oliveto, Massimiliano Di Penta, and Andrea De Lucia. Improving Multi-Objective Test Case Selection by Injecting Diversity in Genetic Algorithms. *IEEE Transactions on Software Engineering*, 41(4):358–383, October 2014.

[PP17] Margherita Peruzzini and Marcello Pellicciari. A framework to design a human-centred adaptive manufacturing system for aging workers. *Advanced Engineering Informatics*, 33(C):330–349, August 2017.

[PSS$^+$16] José A Parejo, Ana B Sánchez, Sergio Segura, Antonio Ruiz-Cortés, Roberto E Lopez-Herrejon, and Alexander Egyed. Multi-objective test case prioritization in highly configurable systems: A case study. *Journal of Systems and Software*, 122(C):287–310, December 2016.

[PT12] Hong-Seok Park and Ngoc-Hien Tran. An autonomous manufacturing system based on swarm of cognitive agents. *Journal of Manufacturing Systems*, 31(3):337–348, July 2012.

[PWAY16] Dipesh Pradhan, Shuai Wang, Shaukat Ali, and Tao Yue. Search-Based Cost-Effective Test Case Selection within a Time Budget: An Empirical Study. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '16, page 1085–1092, New York, NY, USA, July 2016. ACM.

[QES20] Miriam Ugarte Querejeta, Leire Etxeberria, and Goiuria Sagardui. Towards a DevOps Approach in Cyber Physical Production Systems Using Digital Twins. In António Casimiro, Frank Ortmeier, Erwin Schoitsch, Friedemann Bitsch, and Pedro Ferreira, editors, *Computer Safety, Reliability, and Security. SAFECOMP 2020 Workshops*, volume 12235 of *Lecture Notes in Computer Science*, pages 205–216, Cham, September 2020. Springer.

[QTZZ18] Qinglin Qi, Fei Tao, Ying Zuo, and Dongming Zhao. Digital Twin Service towards Smart Manufacturing. In Lihui Wang, editor, *51st CIRP Conference on Manufacturing Systems*, volume 72 of *Procedia CIRP*, pages 237–242. Elsevier, June 2018.

[Ren10] Paolo Renna. Capacity reconfiguration management in reconfigurable manufacturing systems. *The International Journal of Advanced Manufacturing Technology*, 46(1):395–404, January 2010.

[RGC14] Robert Roggio, Jamie Gordon, and James Comer. Taxonomy of common software testing terminology: Framework for key software engineering testing concepts. *Journal of Information Systems Applied Research*, 7(2):4, May 2014.

[RKC$^+$06] Jeanine Romano, Jeffrey D Kromrey, Jesse Coraggio, Jeff Skowronek, and Linda Devine. Exploring methods for evaluating group differences on the NSSE and other surveys: Are the t-test and Cohen'sd indices the most appropriate choices. In *annual meeting of the Southern Association for Institutional Research*, pages 1–51. Citeseer, November 2006.

[Rod17] Blaž Rodič. Industry 4.0 and the New Simulation Modelling Paradigm. *Organizacija. Journal of Management, Informatics and Human Resources*, 50(3):193–207, August 2017.

[RRV19] Aurora Ramirez, José Raúl Romero, and Sebastian Ventura. A survey of many-objective optimisation in search-based software engineering. *Journal of Systems and Software*, 149:382–395, March 2019.

[RVWLB15] Roland Rosen, Georg Von Wichert, George Lo, and Kurt D Bettenhausen. About the importance of autonomy and digital twins for the future of manufacturing. In Alexandre Dolgui, Jurek Sasiadek, and Marek Zaremba, editors, *15th IFAC Symposium on Information Control Problems in Manufacturing*, volume 48 of *IFAC-PapersOnLine*, pages 567–572. Elsevier, December 2015.

[RW07] Gunther Reinhart and Georg Wünsch. Economic application of virtual commissioning to mechatronic production systems. *Production Engineering*, 1:371–379, November 2007.

[SAMW17] Benjamin Schleich, Nabil Anwer, Luc Mathieu, and Sandro Wartzack. Shaping the digital twin for design and production engineering. *CIRP Annals*, 66(1):141–144, April 2017.

[SBL$^+$20] Xuemin Sun, Jinsong Bao, Jie Li, Yiming Zhang, Shimin Liu, and Bin Zhou. A digital twin-driven approach for the assembly-commissioning of high precision products. *Robotics and Computer-Integrated Manufacturing*, 61:101839, February 2020.

[SCD$^+$10] Mike Shafto, Mike Conroy, Rich Doyle, Ed Glaessgen, Chris Kemp, Jacqueline LeMoigne, and Lui Wang. Modeling, Simulation, Information Technology and Processing Roadmap. Technical report, National Aeronautics and Space Administration, May 2010.

[SCN$^+$21] Seung Yeob Shin, Karim Chaouch, Shiva Nejati, Mehrdad Sabetzadeh, Lionel C. Briand, and Frank Zimmer. Uncertainty-aware specification and analysis for hardware-in-the-loop testing of cyber-physical systems. *Journal of Systems and Software*, 171:110813, January 2021.

[SF11] Doaa Soliman and Georg Frey. Verification and Validation of Safety Applications based on PLCopen Safety Function Blocks using Timed Automata in Uppaal. *Control engineering practice*, 19(9):929–946, June 2011.

[SFB$^+$15] Hendrik Simon, Nico Friedrich, Sebastian Biallas, Stefan Hauck-Stattelmann, Bastian Schlich, and Stefan Kowalewski. Automatic test case generation for PLC programs using coverage metrics. In *IEEE 20th Conference on Emerging Technologies & Factory Automation*, ETFA 2015, pages 1–4. IEEE, October 2015.

[SGS18] Luca Sbaglia, Hermes Giberti, and Marco Silvestri. The Cyber-Physical Systems Within the industry 4.0 Framework. In Alessandro Gasparetto Giuseppe Carbone, editor, *Advances in Italian Mechanism Science. IFToMM ITALY*, volume 68 of *Mechanisms and Machine Science*, pages 415–423, Cham, October 2018. Springer.

[SHC$^+$18] Matthias Schamp, Steven Hoedt, Arno Claeys, El-Houssaine Aghezzaf, and Johannes Cottyn. Impact of a virtual twin on commissioning time and quality. In Marco Macchi, László Monostori, and Roberto Pinto, editors, *16th IFAC Symposium on Information Control Problems in Manufacturing*, volume 51 of *IFAC-PapersOnLine*, pages 1047–1052. Elsevier, June 2018.

[Sie15] Siemens AG. *SIMATIC Function Manual, STEP 7 Standards Compliance according to IEC 61131-3*, 3rd edition edition, April 2015.

[SJB16] Jiyoung Song, Eunkyoung Jee, and Doo-Hwan Bae. Automated test sequence generation for function block diagram programs. In *23rd*

Asia-Pacific Software Engineering Conference, APSEC 2016, pages 305–312. IEEE, December 2016.

[SJB18] Jiyoung Song, Eunkyoung Jee, and Doo-Hwan Bae. FBDTester 2.0: Automated test sequence generation for FBD programs with internal memory states. *Science of Computer Programming*, 163:115–137, October 2018.

[SKH18] Radovan Skokan, Martin Krajčovič, and Blanka Horváthová. Industry 4.0 and digital twin. In *Proceedings of the Scientific International Conference InvEnt 1st Edition*, pages 42–45. Žilina, CEIT, 2018.

[SMRM20] WT Seloane, K Mpofu, BI Ramatsetse, and DJPC Modungwa. Conceptual design of intelligent reconfigurable welding fixture for rail car manufacturing industry. In Peter Butala Khumbulani Mpofu, editor, *Enhancing design through the 4th Industrial Revolution Thinking*, volume 91 of *Procedia CIRP*, pages 583–593. Elsevier, January 2020.

[SPAR18] Michael Schluse, Marc Priggemeyer, Linus Atorf, and Juergen Rossmann. Experimentable Digital Twins – Streamlining Simulation-Based Systems Engineering for Industry 4.0. *IEEE Transactions on Industrial Informatics*, 14(4):1722–1731, February 2018.

[STS$^+$19] David Sanderson, Alison Turner, Emma Shires, Jack Chaplin, and Svetan Ratchev. Demonstration of transformable manufacturing systems through the evolvable assembly systems project. Technical report, SAE Technical Paper, January 2019.

[STS$^+$20] David Sanderson, Alison Turner, Emma Shires, Jack C. Chaplin, and Svetan Ratchev. Implementing Large-Scale Aerospace Assembly 4.0 Demonstration Systems. In Rolf Findeisen, Sandra Hirche, Klaus Janschek, and Martin Mönnigmann, editors, *21st IFAC World Congress*, volume 53 of *IFAC-PapersOnLine*. Elsevier, July 2020.

[SWCL17] Rikard Söderberg, Kristina Wärmefjord, Johan S Carlson, and Lars Lindkvist. Toward a digital twin for real-time geometry assurance in individualized production. *CIRP Annals*, 66(1):137–140, 2017.

[SWT12] Markus Simros, Martin Wollschlaeger, and Stefan Theurich. Programming embedded devices in IEC 61131-languages with industrial

PLC tools using PLCopen XML. In *10th Portuguese Conference on Automatic Control*, CONTROLO'2012, July 2012.

[TBSW20] Behrang Ashtari Talkhestani, Dominik Braun, Wolfgang Schloegl, and Michael Weyrich. Qualitative and quantitative evaluation of reconfiguring an automation system using Digital Twin. In Robert X. Gao and Kornel Ehmann, editors, *53rd CIRP Conference on Manufacturing Systems*, volume 93 of *Procedia CIRP*, pages 268–273. Elsevier, April 2020.

[TCZJ17] Ye Tian, Ran Cheng, Xingyi Zhang, and Yaochu Jin. Platemo: A matlab platform for evolutionary multi-objective optimization [educational forum]. *IEEE Computational Intelligence Magazine*, 12(4):73–87, 2017.

[TIES11] Eric J Tuegel, Anthony R Ingraffea, Thomas G Eason, and S Michael Spottswood. Reengineering aircraft structural life prediction using a digital twin. *International Journal of Aerospace Engineering*, 2011, October 2011.

[TJ10] Michael Tiegelkamp and Karl-Heinz John. *IEC 61131-3: Programming industrial automation systems*. Springer, Berlin, Heidelberg, June 2010.

[TJL$^+$19] Behrang Ashtari Talkhestani, Tobias Jung, Benjamin Lindemann, Nada Sahlab, Nasser Jazdi, Wolfgang Schloegl, and Michael Weyrich. An architecture of an Intelligent Digital Twin in a Cyber-Physical Production System. *Automatisierungstechnik*, 67(9):762–782, September 2019.

[TLHN20] Fei Tao, Ang Liu, Tianliang Hu, and A.Y.C. Nee. *Digital Twin Driven Smart Design*, chapter Digital twin based virtual commissioning for computerized numerical control machine tools, pages 289–307. Academic Press, January 2020.

[TLNN18] Ville Toivonen, Minna Lanz, Hasse Nylund, and Harri Nieminen. The FMS Training Center-a versatile learning environment for engineering education. In Mourtzis Dimitris and Chryssolouris George, editors, *Advanced Engineering Education Training for Manufacturing Innovation, 8th CIRP Sponsored Conference on Learning*

Factories, volume 23 of *Procedia Manufacturing*, pages 135–140. Elsevier, April 2018.

[TQWN19] Fei Tao, Qinglin Qi, Lihui Wang, and AYC Nee. Digital Twins and Cyber–Physical Systems toward Smart Manufacturing and Industry 4.0: Correlation and Comparison. *Engineering*, 5(4):653–661, August 2019.

[TSL$^+$19] Fei Tao, Fangyuan Sui, Ang Liu, Qinglin Qi, Meng Zhang, Boyang Song, Zirong Guo, Stephen C-Y Lu, and AYC Nee. Digital twin-driven product design framework. *International Journal of Production Research*, 57(12):3935–3953, February 2019.

[TW20] Behrang Ashtari Talkhestani and Michael Weyrich. Digital Twin of manufacturing systems: a case study on increasing the efficiency of reconfiguration. *Automatisierungstechnik*, 68(6):435–444, June 2020.

[TWE18] Rajeeth Tharma, Roland Winter, and Martin Eigner. An approach for the implementation of the digital twin in the automotive wiring harness field. In Marjanović D., Štorga M., Škec S., Bojčetić N., and Pavković N., editors, *15th International Design Conference*, DESIGN, pages 3023–3032. The Design Society, January 2018.

[TZN19] Fei Tao, Meng Zhang, and AYC Nee. Five-dimension digital twin modeling and its key technologies. In *Digital Twin Driven Smart Manufacturing*, pages 63–81. Elsevier, February 2019.

[UQEESM$^+$21] M. Ugarte-Querejeta, L. Etxeberria-Elorza, G. Sagardui-Mendieta, G. Unamuno-Eguren, and I. Bediaga-Escudero. Virtual commissioning in machine tool manufacturing: A survey from industry. *DYNA*, 96(6):612–619, 2021.

[USL$^+$17] Thomas H-J Uhlemann, Christoph Schock, Christian Lehmann, Stefan Freiberger, and Rolf Steinhilper. The digital twin: Demonstrating the potential of real time data acquisition in production systems. In Joachim Metternich and Rupert Glass, editors, *7th Conference on Learning Factories*, volume 9 of *Procedia Manufacturing*, pages 113–120. Elsevier, December 2017.

[UVH18] Sebastian Ulewicz and Birgit Vogel-Heuser. Increasing system test coverage in production automation systems. *Control Engineering Practice*, 73:171–185, April 2018.

[VB97] N VDW-Bericht. Abteilungsübergreifende projektierung komplexer maschinen und anlagen. *Verein Deutscher Werkzeugmaschinenhersteller, Aachen*, 1997.

[VDJB14] András Vörös, Dániel Darvas, Attila Jámbor, and Tamás Bartha. Advanced saturation-based model checking of well-formed coloured petri nets. *Periodica Polytechnica Electrical Engineering and Computer Science*, 58(1):3–13, April 2014.

[VHDF$^+$14] Birgit Vogel-Heuser, Christian Diedrich, Alexander Fay, Sabine Jeschke, Stefan Kowalewski, Martin Wollschlaeger, et al. Challenges for software engineering in automation. *Journal of Software Engineering and Applications*, 7(5), May 2014.

[VHFST15] Birgit Vogel-Heuser, Alexander Fay, Ina Schaefer, and Matthias Tichy. Evolution of software in automated production systems: Challenges and research directions. *Journal of Systems and Software*, 110:54–84, December 2015.

[VHWK05] Birgit Vogel-Heuser, Daniel Witsch, and Uwe Katzke. Automatic code generation from a UML model to IEC 61131-3 and system configuration tools. In *International Conference on Control and Automation*, volume 2, pages 1034–1039. IEEE, June 2005.

[VK15] Vijay K Vaishnavi and William Kuechler, Jr. *Design Science Research Methods and Patterns: Innovating Information and Communication Technology*. CRC Press, Boca Raton, FL, USA, 2nd edition, May 2015.

[VPS21] Shanu Verma, Millie Pant, and Vaclav Snasel. A comprehensive review on NSGA-II for multi-objective combinatorial optimization problems. *IEEE Access*, 9:57757–57791, April 2021.

[Vya09] Valeriy Vyatkin. The iec 61499 standard and its semantics. *IEEE Industrial Electronics Magazine*, 3(4):40–48, December 2009.

[WAG15] Shuai Wang, Shaukat Ali, and Arnaud Gotlieb. Cost-effective test suite minimization in product lines using search techniques. *Journal of Systems and Software*, 103:370–391, May 2015.

[WBCW20] Andreas Wortmann, Olivier Barais, Benoit Combemale, and Manuel Wimmer. Modeling languages in Industry 4.0: an extended systematic mapping study. *Software and Systems Modeling*, 19:67–94, September 2020.

[WF14] Yi-Chen Wu and Chin-Feng Fan. Automatic test case generation for structural testing of function block diagrams. *Information and Software Technology*, 56(10):1360–1376, October 2014.

[WPUG16a] Bernd W Wirtz, Adriano Pistoia, Sebastian Ullrich, and Vincent Göttel. Business Models: Origin, Development and Future Research Perspectives. *Long Range Planning*, 49(1):36–54, February 2016.

[WPUG16b] Bernd W. Wirtz, Adriano Pistoia, Sebastian Ullrich, and Vincent Göttel. Business Models: Origin, Development and Future Research Perspectives. *Long Range Planning*, 49(1):36–54, February 2016.

[WSKR06] Kristen R. Walcott, Mary Lou Soffa, Gregory M. Kapfhammer, and Robert S. Roos. TimeAware Test Suite Prioritization. In *Proceedings of the 2006 International Symposium on Software Testing and Analysis*, ISSTA ’06, page 1–12, New York, NY, USA, July 2006. ACM.

[WW19] Xi Vincent Wang and Lihui Wang. Digital twin-based WEEE recycling, recovery and remanufacturing in the background of Industry 4.0. *International Journal of Production Research*, 57(12):3892–3902, July 2019.

[WWY$^+$20] Yankai Wang, Shilong Wang, Bo Yang, Lingzi Zhu, and Feng Liu. Big data driven Hierarchical Digital Twin Predictive Remanufacturing paradigm: Architecture, control mechanism, application scenario and benefits. *Journal of Cleaner Production*, 248:119299, March 2020.

[YCJ08] Junbeom Yoo, Sungdeok Cha, and Eunkyung Jee. A verification framework for FBD based software in nuclear power plants. In *15th Asia-Pacific Software Engineering Conference*, pages 385–392. IEEE, December 2008.

[YH07] Shin Yoo and Mark Harman. Pareto Efficient Multi-Objective Test Case Selection. In *Proceedings of the 2007 International Symposium on Software Testing and Analysis*, ISSTA '07, page 140–150, New York, NY, USA, July 2007. ACM.

[YH12] Shin Yoo and Mark Harman. Regression testing minimization, selection and prioritization: a survey. *Software testing, verification and reliability*, 22(2):67–120, March 2012.

[YTYT17] W Yang, Y Tan, K Yoshida, and S Takakuwa. *DAAAM International Scientific Book*, volume 16, chapter Digital twin-driven simulation for a cyber-physical system in Industry 4.0, pages 227–234. DAAAM International, Vienna, 2017.

[ZHL$^+$16] Wei Zheng, Robert M Hierons, Miqing Li, XiaoHui Liu, and Veronica Vinciotti. Multi-objective optimisation for regression testing. *Information Sciences*, 334:1–16, March 2016.

[ZLCX18] Pai Zheng, Tzu-Jui Lin, Chun-Hsien Chen, and Xun Xu. A systematic design approach for service innovation of smart product-service systems. *Journal of Cleaner Production*, 201:657–667, November 2018.

[ZLGH06] G Zhang, R Liu, L Gong, and Q Huang. An analytical comparison on cost and performance among DMS, AMS, FMS and RMS. In Anatoli I. Dashchenko, editor, *Reconfigurable manufacturing systems and transformable factories*, pages 659–673. Springer, January 2006.

[ZQZ$^+$20] Kai Zhang, Ting Qu, Dajian Zhou, Hongfei Jiang, Yuanxin Lin, Peize Li, Hongfei Guo, Yang Liu, Congdong Li, and George Q Huang. Digital twin-based opti-state control method for a synchronized production operation system. *Robotics and Computer-Integrated Manufacturing*, 63(C):101892, June 2020.

[ZS20] Pai Zheng and Abinav Shankar Sivabalan. A generic tri-model-based approach for product-level digital twin development in a smart manufacturing environment. *Robotics and Computer-Integrated Manufacturing*, 64:101958, August 2020.

[ZYC19] Yu Zheng, Sen Yang, and Huanchong Cheng. An application framework of digital twin and its case study. *Journal of Ambient Intelligence and Humanized Computing*, 10(3):1141–1153, June 2019.

[ZZT18] Meng Zhang, Ying Zuo, and Fei Tao. Equipment energy consumption management in digital twin shop-floor: A framework and potential applications. In *15th International Conference on Networking, Sensing and Control*, ICNSC 2018, pages 1–5. IEEE, March 2018.

Appendices

Appendix A

Survey Questionnaire

Virtual Commissioning

We would like to invite you to complete this survey about virtual commissioning and the testing of cyber phsycal systems.

The survey is completely anonymous and it will be used to carry out my PhD at Ideko through Mondragon University.

This project is funded by DiManD Innovative Training Network (ITN). DiManD ITN is an European Training Network (ETN) programme funded by the European Union through the Marie Sktodowska-Curie Innovative Training Networks (H2020-MSCA-ITN-2018) under grant agreement number no. 814078

Many thanks in advance!

Miriam Ugarte
Gorka Unamuno

* Indicates required question

Skip to question 13

Personal details

1. Gender *

Mark only one oval.

◯ Female
◯ Male
◯ Non-binary
◯ Prefer not to say
◯ Other: ____________________

2. Which of the following sectors do you currently work in? *

Mark only one oval.

- ◯ Academia
- ◯ Industry
- ◯ Non-profit
- ◯ Government
- ◯ Other: ______________________

3. Which of the following options describes best your company's activity ? *

Mark only one oval.

- ◯ Metalworking machine tool (chip removal, metal forming, etc.)
- ◯ Flexible manufacturing cells
- ◯ Manufacturing solutions
- ◯ Transfer machines
- ◯ Additive manufacturing machines
- ◯ Continuous process machinery (eg. paper industry)
- ◯ Bespoke machinery
- ◯ Other: ______________________

4. Job position *

____

5. Which of the following fields is closest to your formation? *

Mark only one oval.

- ◯ Mechanic
- ◯ Electronic
- ◯ Mechatronic
- ◯ Industrial Design
- ◯ Telecommunications
- ◯ Information and Communications Technology (ICT)
- ◯ Informatics
- ◯ Automation
- ◯ Business management and administration
- ◯ Industrial organisation
- ◯ Other: ____________________

6. Work experience *

Mark only one oval.

- ◯ < 5 years
- ◯ 5 - 9 years
- ◯ 10 - 20 years
- ◯ > 20 years

7. Work experience in machine tool industry *

Mark only one oval.

- ◯ < 5 years
- ◯ 5 - 9 years
- ◯ 10 - 20 years
- ◯ > 20 years

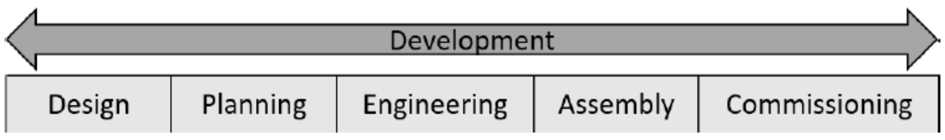
8. Which of the following technologies and tools are closest to your daily activities? *

Tick all that apply.

- ☐ Operation Technologies (OT): PLC, SCADA, DCS, CNC, etc.
- ☐ Scientific instrument: oscilloscope, spectrum analyzer, multimeter, etc.
- ☐ 3D simulators/emulators: Siemens NX, Delmia, ISG, Simumatik, Demo3D, Experior, Visual Components, etc.
- ☐ UML tools: Enterprise Architect, Microsoft Visio, etc.
- ☐ CAD/CAM systems: AutoCAD, SOLIDWORKS, etc.
- ☐ ERP/MRP/MES systems: SAP, Oracle, Microsoft Dynamics, Salesforce, etc.
- ☐ Numerical computation: Lab View, MathCAD, MAPLE, Matlab, etc.
- ☐ Big Data: Spark, Kafka, Splunk, Hadoop, Tableau, Grafana, etc.
- ☐ IoT platforms: MindSphere, Thingworx, Predix, etc.
- ☐ Database: Sql, NoSql, Oracle, MySQL, SQLite, MongoDB, Cassandra, etc.
- ☐ Web technology: CSS3, HTML5, JS, PHP, Jquery, CMS, Restful API, Angular, etc.
- ☐ Visual programming: NodeRed, Visuino, Embrio, Ardublock, etc.
- ☐ Data analytics: Python, R, etc.
- ☐ Cloud computing: Amazon Web Services, Google Compute Engine, Microsoft Azure, OpenStack, etc.
- ☐ Containers: Docker, Kubernetes, etc.
- ☐ Test automation: JUnit, TestNG, Robot Framework, etc.
- ☐ Project management: Jira, Confluence, Basecamp, etc.
- ☐ Control version: Git, Apache Subversion (SVN), etc.
- ☐ None of the aforementioned tools

Traditional Commissioning

Commissioning is the last stage of the product development process. In this phase the engineers must guarantee the correct operation of the system.



9. What are the main challenges in traditional Commissioning?

Tick all that apply.

- ☐ Time to market
- ☐ Little margin for error correction
- ☐ Unexpected issues due to errors from previous development stages (design, engineering, etc.)
- ☐ Difficulties in testing the electrical system
- ☐ Development cost
- ☐ Others (please specify below):
- ☐ Other: ______________________________

10. Who participates during the commissioning stage?

Tick all that apply.

- ☐ Mechanical engineers
- ☐ Electronic engineers
- ☐ CNC control and automata engineers
- ☐ SW engineers
- ☐ Robotics engineers
- ☐ Telecommunications engineers
- ☐ Systems engineers
- ☐ Other: ______________________________

11. How long does the commissioning stage take?

Mark only one oval.

	Week(s)
1	◯
2	◯
3	◯
4	◯
5	◯
6	◯
7	◯
8	◯
9	◯
10	◯

12. Please indicate the degree of complexity of the commissioning stage.

Mark only one oval.

Less complex

0 ◯

1 ◯

2 ◯

3 ◯

4 ◯

5 ◯

6 ◯

7 ◯

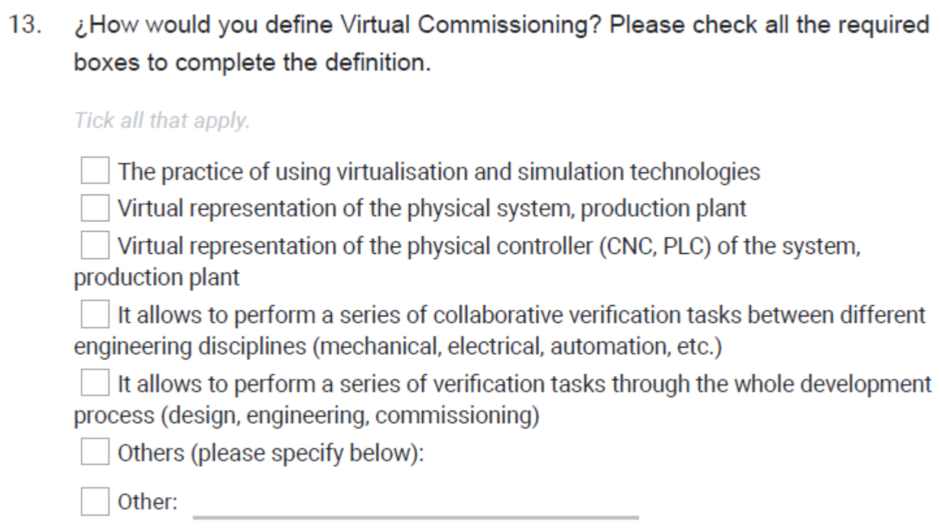
8 ◯

9 ◯

10 ◯

More complex

Virtual Commissioning



13. ¿How would you define Virtual Commissioning? Please check all the required boxes to complete the definition.

Tick all that apply.

- ☐ The practice of using virtualisation and simulation technologies
- ☐ Virtual representation of the physical system, production plant
- ☐ Virtual representation of the physical controller (CNC, PLC) of the system, production plant
- ☐ It allows to perform a series of collaborative verification tasks between different engineering disciplines (mechanical, electrical, automation, etc.)
- ☐ It allows to perform a series of verification tasks through the whole development process (design, engineering, commissioning)
- ☐ Others (please specify below):
- ☐ Other: ____________________

14. Have you implemented Virtual commissioning? Please indicate the degree of implementation.

Mark only one oval.

0	◯
1	◯
2	◯
3	◯
4	◯
5	◯
6	◯
7	◯
8	◯
9	◯
10	◯

15. Could you specify the software or the tool that has been used?

16. What type of Virtual Commissioning has been realised?

Tick all that apply.

- ☐ Hardware in the loop: real controler and virtual system/production plant
- ☐ Reality in the loop: virtual controller and real system/production plant
- ☐ Software in the loop: virtual controller and virtual system/production plant

17. Who participates during the process of Virtual Commissioning ?

Tick all that apply.

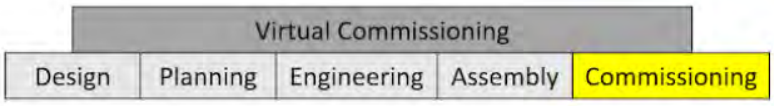
- ☐ Mechanical engineers
- ☐ Electronic engineers
- ☐ CNC control and automata engineers
- ☐ SW engineers
- ☐ Robotics engineers
- ☐ Telecommunications engineers
- ☐ Systems engineers
- ☐ Other: ______________

18. At which of the following stages of the development process is Virtual Commissioning carried out?

Mark only one oval per row.

	%0	%25	%50	%75	%100
Design	◯	◯	◯	◯	◯
Modelling	◯	◯	◯	◯	◯
Engineering	◯	◯	◯	◯	◯
Electrical Planning	◯	◯	◯	◯	◯
Assembly	◯	◯	◯	◯	◯
Commissioning	◯	◯	◯	◯	◯

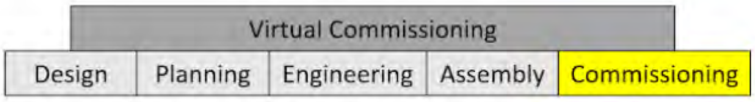
19. How long does the Commissioning stage (the last phase of the development process) take when carrying out virtual commissioning?



Mark only one oval.

	Week(s)
1	○
2	○
3	○
4	○
5	○
6	○
7	○
8	○
9	○
10	○

20. Please indicate the degree of complexity of the Commissioning stage (the last phase of the development process) take when carrying out virtual commissioning



Mark only one oval.

Less complex

0 ◯

1 ◯

2 ◯

3 ◯

4 ◯

5 ◯

6 ◯

7 ◯

8 ◯

9 ◯

10 ◯

More complex

21. ¿What has been currently performed with Virtual Commissioning or what is the ultimate goal?

22. What are the main challenges faced with Virtual commissioning? or what are the main barriers that prevent from achieving success stories?

23. What are the main benefits that Virtual Commissioning has brought against the traditional Commissioning?

Digital Twin

24. ¿How would you define the Digital Twin? Please check all the required boxes to complete the definition.

Tick all that apply.

☐ The practice of using virtualisation and simulation technologies

☐ Virtual representation of a physical asset

☐ Virtual representation of a physical asset, process, human, performance, etc.

☐ It is composed of three components: physical entity, virtual model, connection between both counterparts

☐ It is composed of five components: physical entity, virtual model, data, services, connections

☐ The physical asset is dynamically updating as the virtual model changes

☐ The virtual model is dynamically updating as the physical asset changes

☐ It requires real time communication between the virtual model and the physical asset

☐ Others (please specify below):

☐ Other: ______________________

25. Have you implemented the Digital Twin? Please indicate the degree of implementation.

Mark only one oval.

0	◯
1	◯
2	◯
3	◯
4	◯
5	◯
6	◯
7	◯
8	◯
9	◯
10	◯

NO, it has not been implemented yet

26. Do you think that the use of the Digital Twin would bring any benefit?

Mark only one oval.

- ◯ Yes
- ◯ No

27. Why?

YES, It has been implemented (with the degree of implementation indicated above)

28. ¿What has been the main reason for the implementation of the Digital Twin or what is the ultimate goal?

29. What does the implemented digital twin represent?

Tick all that apply.

- ☐ Controller (CNC, PLC) of the real system, production line
- ☐ Real system
- ☐ Robot, machine
- ☐ Production line
- ☐ Process of the system
- ☐ Performance of the system
- ☐ Others (please specify below):
- ☐ Other: ____________

30. Could you specify the software or the tool that has been used?

____

31. What are the main challenges faced with the Digital Twin? or what are the main barriers that prevent from achieving success stories?

____

32. What are the main benefits that has brought the Digital twin?

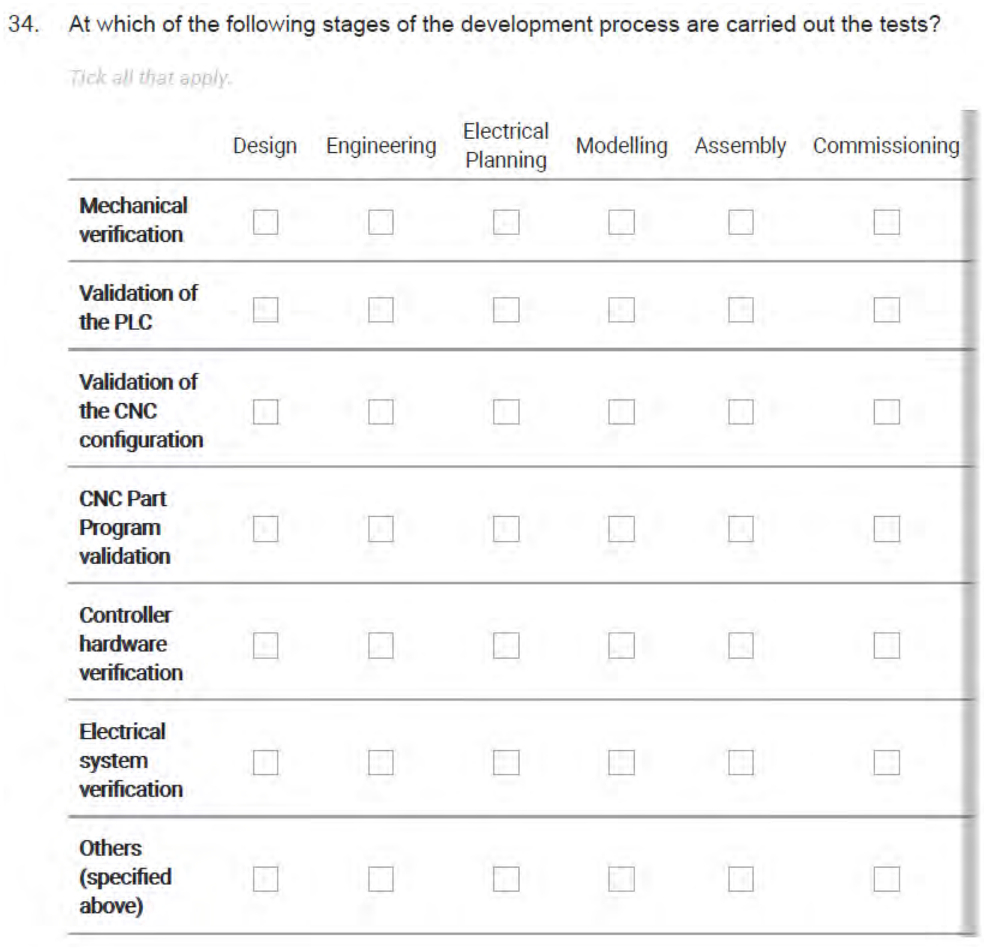
____

Machine tool - Verification and Validation tests

33. Which of the following tests are carried out?

Tick all that apply.

☐ Mechanical verification
☐ Validation of the PLC (control logic, optional subsystem integration, vertical integration with MES system)
☐ Validation of the CNC configuration (test axle parameters, transformation functions, "M" codes)
☐ CNC Part Program validation (G-Code, Cycle time, Tool Path verification),
☐ Controller hardware verification: input/output signals, input/output interfaces, communication bus, etc.
☐ Electrical system verification
☐ Others (please specify below):
☐ Other: ______________________



34. At which of the following stages of the development process are carried out the tests?

Tick all that apply.

	Design	Engineering	Electrical Planning	Modelling	Assembly	Commissioning
Mechanical verification	☐	☐	☐	☐	☐	☐
Validation of the PLC	☐	☐	☐	☐	☐	☐
Validation of the CNC configuration	☐	☐	☐	☐	☐	☐
CNC Part Program validation	☐	☐	☐	☐	☐	☐
Controller hardware verification	☐	☐	☐	☐	☐	☐
Electrical system verification	☐	☐	☐	☐	☐	☐
Others (specified above)	☐	☐	☐	☐	☐	☐

35. How are the tests realised?

Tick all that apply.

	Manual test	with the help of a SW or a tool	Automated test
Mechanical verification	☐	☐	☐
Validation of the PLC	☐	☐	☐
Validation of the CNC configuration	☐	☐	☐
CNC Part Program validation	☐	☐	☐
Controller hardware verification	☐	☐	☐
Electrical system verification	☐	☐	☐
Others (specified above)	☐	☐	☐

36. Do you think that automated tests would bring any benefit?

Mark only one oval per row.

	Yes	No
Mechanical verification	◯	◯
Validation of the PLC	◯	◯
Validation of the CNC configuration	◯	◯
CNC Part Program validation	◯	◯
Controller hardware verification	◯	◯
Electrical system verification	◯	◯
Others (specified above)	◯	◯

37. Why?

38. What are the main challenges faced when carrying out the aforementioned tests? or what are the main barriers that prevent from achieving success stories?

39. What are the main challenges faced in the development/operations of a machine tool?

Mark only one oval per row.

	%0	%25	%50	%75	%100
Controller HW problems	◯	◯	◯	◯	◯
Controller SW problems	◯	◯	◯	◯	◯
Simulator/emulator SW problems	◯	◯	◯	◯	◯
PLC problems	◯	◯	◯	◯	◯
Mechatronic system problems	◯	◯	◯	◯	◯
Communication problems	◯	◯	◯	◯	◯
Integration problems	◯	◯	◯	◯	◯
Execution time, standby time, etc.	◯	◯	◯	◯	◯
Lack of precision	◯	◯	◯	◯	◯
Lack of stability	◯	◯	◯	◯	◯
Unexpected circunstances	◯	◯	◯	◯	◯
Unexpected issues due to errors from previous development stages (design, engineering, etc.)	◯	◯	◯	◯	◯

Appendix B

Ethics Requirements

Annex 1. Template of the informed consent form for interviews

In this section, the contents of the templates that will be used in each project action including human participants are presented. Participants are considered as co-researchers in DiManD, as stated elsewhere.

The drafts on the above mentioned issues are presented, including the main points related to guaranteeing informed consent of all the stakeholders.

Information sheets

The contents of the information sheets that will be used in each one of the activities within DiManD entailing human participation are presented. This information is necessary to obtain proper informed consent.

Information Document:

- *Why the person has been invited to participate*
- *What he/she will have to do if he/she decides to take part*
- *What will be the risks of taking part of it*
- *How to withdraw from the project*
- *What we will do with their DATA*

Information sheet on Protection of Personal Data (POPD)

Dear participant,

You are willing to participate in DiManD, a training framework for Europe with the purpose of improving Europe's industrial competitiveness by designing and implementing an integrated programme in the area of intelligent informatics driven manufacturing that will form the benchmark for training future Industrie 4.0 practitioners.

Before taking part, and having understood the purpose of the project explained in this information document, it is essential to us that you understand which personal data we are collecting and how we will be treating this information.

Please, read this document carefully. If you have any doubt or something is unclear to you, please ask any of the project partners, face-to-face when possible, or by email at dimand.mgep@mondragon.edu

Purpose of the project

The Digital Manufacturing and Design (DiManD) Innovative Training Network (ITN) is a European Training Network (ETN) programme that will provide high-quality multidisciplinary, multi-professional and cross-sectorial research and training to high-achieving early stage researchers in the area of **Industrie 4.0**. DiManD comprises a well-balanced consortium that spans six European countries and

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant No. 814078

incorporates academic and industry sectors to promote international, interdisciplinary and inter-sectoral aspects of ESR skill development.

DiManD aims to develop a high-quality multidisciplinary, multi-professional and cross-sectorial research and training framework for Europe with the purpose of improving Europe's industrial competitiveness by designing and implementing an integrated programme in the area of intelligent informatics driven manufacturing that will form the benchmark for training future Industrie 4.0 practitioners.

Why have I been invited to participate?

You have been invited to participate to DiManD: [1] as a member of one of the participants in the project (Beneficiary, Partner Organization, Supervisor, ESR), [2] or as an expert in the area of industrial digitalization.

What will I have to do if I decide to take part?

You can contribute, as the other researchers, sharing your knowledge, experience and opinions through surveys.

As for your ethical duties, you will not behave in an unethical way, that is, you will not communicate fake information with bad faith.

What will I have to do if I decide to take part?

Regarding potential risks, , those does not exceed in probability or magnitude the ones that could be expected in a work activity based on agreed meetings (physical and/or online), in which experiences and knowledge are discussed and shared around common projects.

Participation in this project is entirely voluntary and unpaid. Meals are covered as well as travel expenses when it is necessary to move from one city to another.

How to withdraw from the project?

You can withdraw from the project anytime you decide it, by email to dimand.mgep@mondragon.edu or by mail to Mondragon Goi Eskola Politeknikoa, S.Coop. Loramendi 4, 20500 Arrasate Gipuzkoa (Spain).

You do not need to add any explanation of why do you want to withdraw from the project.

What we will do with your DATA?

All personal data is regulated under the General Data Protection Regulation (EU) 2016/679 (GRPD). You can erase all your personal data from all project platforms at any time. You can also download all the data that you have provided to the project. You can also be part of audio/video recording or photographs which might be used only to document and disseminate the project activities, if you specifically consent.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant No. 814078

Your personal data will be used only for:

If it is previously accepted, contact you to inform you about new events specifically related to DiManD, or advances on the project.

Who is responsible for data processing?

MGEP is the responsible for processing all personal - and no personal – data obtained. MGEP is a non-profit private foundation and the coordinator of the DiManD project.

Personal data transfer

We follow strict security procedures when storing personal data. Under no circumstances we will transfer personal data to third parties.

Rights and how to exercise them. You have the right to:

- Request information about whether we have personal data about you and, if so, what information we have, why we have it and how we are using it.
- Request access to your personal data. This allows you to receive a copy of your personal data and to correct any incomplete or incorrect information.
- Request personal data deletion. This allow you to delete or ask us to delete your personal data.
- Request to transfer your personal data to you or to a third party in an electronic and structured format - commonly known as the right to data portability.
- You can exercise any of these rights by email to dimand.mgep@mondragon.edu
- You will not have to pay any fee to access your personal data - or to exercise any other of your rights

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant No. 814078

Informed consent/assent forms

The contents of the informed consent/assent forms that will be used in each of the activities in DiManD, both online and offline, are presented as follows:

INFORMED CONSENT FORM

Project acronym	DiManD
Project name	Digital Manufacturing and Design (DiManD)
Grant Agreement no.	814078
Project type	European Training Network (ETN) programme funded by the European Union through the Marie Sktodowska-Curie Innovative Training Networks (H2020-MSCA-ITN-2018)
Start date of the project	01/05/2019
End date of the project	31/04/2023
This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 814078.	
Disclaimer: The views and opinions expressed in this document are solely those of the project, not those of the European Commission.	

Introduction

Before making a decision on whether you want to participate or not, please read this document carefully. Please feel free to ask any questions to ensure that you fully understand the purpose and proceedings of this study, including risks and benefits. As the study is carried out in a language that is not your mother tongue, this informed consent document may include words that you do not understand. If this is the case, please ask the interviewer to fully explain the meaning of the word or piece of information you do not fully understand.

Compliance with legal and ethical regulations

We assure full compliance with relevant legislation on data protection and ethical standards.

Study result

Your participation in the study will feed the research results of DiManD project. A report will be drafted, which will be supplied to you.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant No. 814078

Consent

By signing this document, you are agreeing to take part in the study. You will be given a copy of this document for your records and one copy will be kept by the project coordinator with the study records. Be sure that questions you have about the study have been answered and that you understand what you are being asked to do. You may contact the researcher if you think of a question later.

I agree to participate in the study.

This consent form is made pursuant to the relevant national, European and international data protection laws and regulations and personal data treatment obligations. Specifically this consent document complies with the EC Data Protection Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data.

…………………………………………………………………………………………

Name and surname of participant

…………………………………………………………………………………………

Place, date and signature of participant

Statement of investigator's responsibility: I have explained the nature and purpose of this research study, the procedures to be undertaken and any risks that may be involved. I have offered to answer any questions and fully answered such questions. I believe that the participant understands my explanation and has freely given informed consent.

…………………………………………………………………………………………

Name and surname of the researcher

…………………………………………………………………………………………

Place, date and signature of the researcher

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant No. 814078

Appendix C

Code

C.1 TIAtestSuiteConverter.java

```
import java.io.*;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
import javax.print.attribute.Size2DSyntax;




public class TIAtestSuiteConverter{


 public static void main(String[] args) throws IOException
 {

 String usecase = "frtd";
 int mutants = 1;
 boolean first = true;
 File dir = null;

 for(int n=0;n<mutants+1;n++) {

 if(first){
 dir = new File("E:\\"+usecase+"\\results");
 first=false;
 }
String[] fileNames = dir.list();


 for (String fileName : fileNames) {

 String file1="E:\\"+usecase+"\\results\\"+fileName;
 BufferedReader br1 = new BufferedReader(new FileReader(file1));
```

```
String[] inouts=null;
String[] values=null;
List<ArrayList<String>> inputs = new ArrayList<ArrayList<String>>();
ArrayList<ArrayList<String>> ivalues = new ArrayList<ArrayList<String>>();
ArrayList<ArrayList<String>> outputs = new ArrayList<ArrayList<String>>();
ArrayList<ArrayList<String>> ovalues = new ArrayList<ArrayList<String>>();
ArrayList<String> inputsArray = new ArrayList<String>();
ArrayList<String> ivaluesArray = new ArrayList<String>();
ArrayList<String> outputsArray = new ArrayList<String>();
ArrayList<String> ovaluesArray = new ArrayList<String>();

String line = br1.readLine();
String key = null;

while (line!= null)
{
if(line.isEmpty())
line=br1.readLine();

else{
inouts=line.substring(1, line.length()-1 ).split(";");

values=inouts[1].split(" ");

String newkey=values[0];

if(newkey.contains("_t"))
newkey=newkey.substring(0, newkey.lastIndexOf("_t"));


if(key==null)
key=newkey;

if(!newkey.equals(key)){
key=newkey;
if(!inputsArray.isEmpty()){
inputs.add(new ArrayList<String>(inputsArray));
ivalues.add(new ArrayList<String>(ivaluesArray));
inputsArray.clear();
  ivaluesArray.clear();
}
else if (!outputsArray.isEmpty()){
outputs.add(new ArrayList<String>(outputsArray));
ovalues.add(new ArrayList<String>(ovaluesArray));
outputsArray.clear();
ovaluesArray.clear();
}
}

if(inouts[0].equals("input")){
inputsArray.add(values[0]);
ivaluesArray.add(values[1]);
}
```

```
 else if(inouts[0].equals("output")) {
 outputsArray.add(values[0]);
 ovaluesArray.add(values[1]);
 }
 line=br1.readLine();
 }
}

 // last item

 if(!inputsArray.isEmpty()){
 inputs.add(new ArrayList<String>(inputsArray));
 ivalues.add(new ArrayList<String>(ivaluesArray));
 inputsArray.clear();
   ivaluesArray.clear();
 }
 else if (!outputsArray.isEmpty()){
 outputs.add(new ArrayList<String>(outputsArray));
 ovalues.add(new ArrayList<String>(ovaluesArray));
 outputsArray.clear();
 ovaluesArray.clear();
 }

 //write TIA Test Suite solution
 for(int i=0;i<inputs.size();i++)
 mapp(inputs.get(i));

 for(int i=0;i<outputs.size();i++)
 mapp(outputs.get(i));

 if(!outputs.isEmpty()){

 for(int k=0;k<outputs.get(0).size();k++){
 String fname_tc=fileName.substring(0,fileName.length()-4);
 String tc=fname+"_t"+(outputs.get(0).size()-k-1);

 File outputDir=new File("E:FBDTester\\output\\"+usecase+"\\TIAtestsuite");

if (!outputDir.exists())
outputDir.mkdirs();
 String fout="E:FBDTester\\output\\"+usecase+"\\TIAtestsuite\\"+tc+".tat";
 PrintWriter pw = new PrintWriter(fout);
 String input = null;
 String output = null;
 Pattern pattern = Pattern.compile("\".\"[0-9]");
 Matcher matcher;
 int start=0;
 int end=0;
 int t=0;
 String iter=null;

 pw.println("TEST_CASE \""+tc+"\"");
pw.println("PROPERTY");
pw.println("SCOPE : \"PLC_1\"");
pw.println("END_PROPERTY");

pw.println("VAR");


for(int i=0;i<inputs.size();i++) {
for(int j=0;j<inputs.get(i).size();j++){
input = inputs.get(i).get(j);
if(inputs.get(i).size()>1){
if(input.contains("_t")){
iter=input.substring(input.lastIndexOf("_t")+2,input.length()-1);
  t=Integer.parseInt(iter);
  pw.print(input+":");

  input=input.replace(".", "\".\"");
  matcher=pattern.matcher(input);
  while(matcher.find()) {
 start=matcher.start();
 input=input.substring(0,start)+"."+input.substring(start+3);
  }

  pw.print(input.substring(0,input.lastIndexOf("_t")));
pw.println("\".INPUT["+(inputs.get(i).size()-1-t)+"];");
  }
  else {
  t=0;
  pw.print(input+":");
  input=input.replace(".", "\".\"");
  matcher=pattern.matcher(input);
  while(matcher.find()) {
 start=matcher.start();
 input=input.substring(0,start)+"."+input.substring(start+3);
  }
  pw.println(input+".INPUT["+(inputs.get(i).size()-1-t)+"];");
  }
}
else {
if(input.contains("_t")){
  pw.print(input+":");
  input=input.replace(".", "\".\"");
  matcher=pattern.matcher(input);
  while(matcher.find()) {
 start=matcher.start();
 input=input.substring(0,start)+"."+input.substring(start+3);
  }
  pw.println(input.substring(1,input.lastIndexOf("_t"))+";");
  }
  else {
  pw.print(input+":");
  input=input.replace(".", "\".\"");
  matcher=pattern.matcher(input);
  while(matcher.find()) {
 start=matcher.start();
 input=input.substring(0,start)+"."+input.substring(start+3);
  }
  pw.println(input+";");
  }
}

}

}
pw.flush();

pw.println("END_VAR\n");

String value = null;


pw.println("\nSTEP: \""+ (k+1) +" cycles\"\n");

for(int i=0;i<inputs.size();i++) {

for(int j=0;j<inputs.get(i).size();j++){
input = inputs.get(i).get(j);
value = ivalues.get(i).get(j);
  pw.println(input+":="+value+";");
}

}

pw.println("\nrun(CYCLES:="+(k+2)+");\n");

for(int i=0;i<outputs.size();i++) {
output=outputs.get(i).get(outputs.get(i).size()-k-1);

if(output.contains("_t"))
output=output.substring(1,output.lastIndexOf("_t"));
else
output=output.substring(1,output.length()-1);

value=ovalues.get(i).get(ovalues.get(i).size()-k-1);
if(output.contains("TON"))
pw.println("ASSERT.Equal("+output+",T#"+value+"ms);");
else
pw.println("ASSERT.Equal("+output+","+value+");");
}
pw.println("\nEND_STEP");
pw.println("END_TEST_CASE");
pw.flush();
}

}
}
```

```
}
 }

private static void mapp(ArrayList<String> inouts) {
// TODO Auto-generated method stub

for (int i=0;i<inouts.size();i++) {
if(inouts.get(i).contains("E_SWCamMinus_22"))
inouts.set(i,inouts.get(i).replace("E_SWCamMinus_22", "E_SWCamMinus[22]"));
if(inouts.get(i).contains("E_SWCamPlus_22"))
inouts.set(i,inouts.get(i).replace("E_SWCamPlus_22", "E_SWCamPlus[22]"));
inouts.set(i,"\""+inouts.get(i)+"\"");
}

}
}
```

C.2 generateRandomTestSuites(int maxRun,String testDoc,float coverageGoal)

```
static void generateRandomTestSuites(int maxRun, String testDoc, float coverageGoal)
throws IOException {
String pre = "";
boolean includeAll = false;
if (CreateGUI.BCTestCheck.isSelected()) {
pre += "BC_";
}
else if (CreateGUI.ICCTestCheck.isSelected()) {
pre += "ICC_";
}
else if (CreateGUI.CCCTestCheck.isSelected()) {
pre += "CCC_";
}
else if (CreateGUI.RTestCheck.isSelected()) {
pre = "RT_";
includeAll = true;
coverageGoal = 2; // Cannot be achieved!
}

loadProgramInfoFile();

float coverage;
String log = "No.\tCov.\tSize\tMaxRun\r\n";

int iter=0;
int ncycles=2;
String usecase="LAUNCHER_KAIST";
for (int i = 0; i < maxRun; i++) {

if(iter==5){
ncycles++;
iter=0;
}

coverage = generateRandomTestSuite(usecase, ncycles, testDoc,
coverageGoal, includeAll);
log += i + "\t" + coverage + "\t" + testSuiteSize + "\t";
log+= iterationCount + "\r\n";

iter++;
}


try {
String f="output\\"+ pre + "R-Suite_coverage_levels.txt";
BufferedWriter logFile = new BufferedWriter(new FileWriter(f));
logFile.write(log.trim());
logFile.close();
String dirExec="C:\\Windows\\System32\\notepad.exe output\\"
dirExec+= pre + "R-Suite_coverage_levels.txt"
Runtime.getRuntime().exec(dirExec);
} catch (IOException e) {
e.printStackTrace();
System.exit(-1);
}

testSuiteID = 1;
}
```

C.3 generateRandomTestSuite(String usecase, int maxSize, String testDoc, float coverageGoal, boolean includeAll)

```
private static float generateRandomTestSuite(String usecase, int maxSize,
String testDoc, float coverageGoal, boolean includeAll) {
DPaths.clear();
ICC_DPaths.clear();
CCC_DPaths.clear();
DPCMacros.clear();
functionDPCs.clear();
functionBlockLocalVars.clear();
functionBlockPreVars.clear();

findDataPaths();
sortDataPaths();
calculateDPC();

CreateGUI.window.repaint();
CreateGUI.window.setVisible(true);

ArrayList<ArrayList<Item>> testSuite = new ArrayList<ArrayList<Item>>();
```

```
ArrayList<Item> testCase;

ArrayList<TestCase> testSet = new ArrayList<TestCase>();

//String testcase = "";
ArrayList<String> testcase = new ArrayList<String>();

int iter=0;
int counter=1;

//
//
try {
//     testDoc.txt   .
//      .
//loadTestFile(testDoc);  ,      .
BufferedReader testFile = new BufferedReader(new FileReader(testDoc));

testCase = new ArrayList<Item>();
List<Item> testCCases = new ArrayList<Item>();

int mode = -1;
String thisLine = "";
while ((thisLine = testFile.readLine()) != null) {
if (thisLine.trim().length() == 0)
continue;
if (thisLine.startsWith("//"))
continue;
if (thisLine.startsWith("###")) {
if (thisLine.contains("constants")) {
mode = 1;
} else if (thisLine.contains("inputs")) {
mode = 2;
} else if (thisLine.contains("outputs")) {
mode = 3;
} else if (thisLine.contains("number of test cases")) {
mode = 4;
} else if (thisLine.contains("test cases")) {
mode = 5;
} else if (thisLine.contains("cTypes")){
mode = 6;
}
} else {
switch (mode) {
case 1:
String[] cNames = thisLine.split(", ");
for (int i = 0; i < cNames.length; i++)
testCCases.add(i, new Item(cNames[i]));
break;
case 2: // inputs
String[] inputNames = thisLine.split(", ");
for (int i = 0; i < inputNames.length; i++)
testCase.add(i, new Item(inputNames[i]));
```

```
break;
case 3:
break;
case 4:
break;
case 5: // test cases
String[] testValues = thisLine.split("\t");
if (testValues.length != testCase.size()) {
System.err.println("ERROR: #input != #test-values");
System.exit(-1);
}
for (int i = 0; i < testCase.size(); i++) {
for (Element var : invars) {
if (testCase.get(i).getName().equals(var.invar.getExpression())) {
if (Character.isLetter(testValues[i].charAt(0))) {
var.valueType = var.BOOLEAN;
testCase.get(i).setType(0);
} else if (testCase.get(i).getName().contains("CNT")) {
var.valueType = var.INTEGER;
testCase.get(i).setType(1);
} else {
var.valueType = var.REAL;
testCase.get(i).setType(2);
}

//  random
testCase.get(i).random();

if (testCase.get(i).getType() == 0) {
var.value = (testCase.get(i).getValue() == 0) ? "false" : "true";
} else {
var.value = testCase.get(i).getValue() + "";
}
}
}
}
break;
case 6:
String[] testCValues = thisLine.split("\t");
if (testCValues.length != testCCases.size()) {
System.err.println("ERROR: #input != #test-Cvalues");
System.exit(-1);
}
for (int i = 0; i < testCCases.size(); i++) {
for (Element var : invars) {
if (testCCases.get(i).getName().equals(var.invar.getExpression())) {
if (Character.isLetter(testCValues[i].charAt(0))) {
var.valueType = Element.BOOLEAN;
testCCases.get(i).setType(0);
} else if (testCCases.get(i).getName().contains("CNT")) {
var.valueType = Element.INTEGER;
testCCases.get(i).setType(1);
} else {
```

```
var.valueType = Element.REAL;
testCCases.get(i).setType(2);
}

if (testCCases.get(i).getType() == 0) {
var.value = (testCCases.get(i).getValue() == 0) ? "false" : "true";
} else {
var.value = testCCases.get(i).getValue() + "";
}
}
}
}
break;
} // end of switch
} // end of else
} // end of while

testSuite.add(testCase); // !! IMPORTANT

for (ArrayList<Item> tc : testSuite) {
String result = "";
String value = "";
for (Item item : tc) {
if (item.getType() == 0) {
value = (item.getValue() == 0) ? "false" : "true";
} else {
value = item.getValue() + "";
}
testcase.add("(="+item.getName()+"_t"+(maxSize-1)+" "+ " "+value+")");
}
}

iter++;

testFile.close();
} catch (FileNotFoundException e) {
e.printStackTrace();
System.err.println("randomTestCaseGeneration: FATAL ERROR, file not found");
System.exit(-1);
} catch (IOException e) {
e.printStackTrace();
System.err.println("randomTestCaseGeneration: FATAL ERROR, IOException");
System.exit(-1);
}

// * Test Documentation Load Complete
// * Type   ICC, CCC path
if (CreateGUI.ICCTestCheck.isSelected())
createICCPath();
if (CreateGUI.CCCTestCheck.isSelected())
createCCCPath();

List<DPath> dPath = new ArrayList<DPath>();
```

```
if (CreateGUI.BCTestCheck.isSelected())
dPath.addAll(DPaths);
if (CreateGUI.ICCTestCheck.isSelected())
dPath.addAll(ICC_DPaths);
if (CreateGUI.CCCTestCheck.isSelected())
dPath.addAll(CCC_DPaths);

float coverage = 0, newCoverage = 0;
coverage = assessCoverageLevel(testSuite, dPath);
//System.out.println("1th generation: " + coverage);

int iteration = 2;

while (coverage < coverageGoal && iteration <= maxSize) {
// *  random testCase
testCase = new ArrayList<Item>();
for (Item item : testSuite.get(0)) {
testCase.add(new Item(item.getName(), item.getType()));
}

String result = "";
String value = "";

for (Item item : testCase) {
if (item.getType() == 0) {
value = (item.getValue() == 0) ? "false" : "true";
} else {
value = item.getValue() + "";
}

if(maxSize-iter-1>0)
testcase.add("(="+item.getName()+"_t"+(maxSize-iter-1)+" "+" "+value+")");
else
testcase.add("(=" + item.getName()+" " + " " + value + ")");
}

iter++;

testSuite.add(testCase);



// *  testSuite  coverage
newCoverage = assessCoverageLevel(testSuite, dPath);
//System.out.println(iteration + "th generation: " + newCoverage);

// *  coverage    testSuite
//if (!includeAll && newCoverage <= coverage)
// testSuite.remove(testCase);

coverage = newCoverage;
iteration++;
}
```

```
String writerLog = "";


for (ArrayList<Item> tc : testSuite) {
String result = "";
String value = "";
for (Item item : tc) {
if (item.getType() == 0) {
value = (item.getValue() == 0) ? "false" : "true";
} else {
value = item.getValue() + "";
}
result += "(" + item.getName() + " " + value + ")";
}
writerLog += result + "\n";
}

Collections.sort(testcase);

TestCase tc = new TestCase(String.join("", testcase));
tc.partNo = 1;
tc.counter = counter;
testSet.add(tc);

counter++;
numOfEachTC[1][counter-2] = maxSize-2;

String s_id=String.format("%03d", testSuiteID)+/*"-"+randomSeed+*/".txt";
solutionWriter(testSet, "/"+usecase+"/testcases/RT_based_testSuite_" + s_id);
testSuiteID++;

String pre = "";
if (CreateGUI.BCTestCheck.isSelected()) {
pre += "BC_";
}
if (CreateGUI.ICCTestCheck.isSelected()) {
pre += "ICC_";
}
if (CreateGUI.CCCTestCheck.isSelected()) {
pre += "CCC_";
}
if (CreateGUI.RTestCheck.isSelected())
pre = "RT_";

CreateGUI.console_println("Random Test Generation ... done.");
console_flush();
testSuiteSize = testSuite.size();
iterationCount = iteration-1;
return coverage;
}
```

Appendix D

Exploitation of Results

D.1 Market needs

The main market drivers are highlighted below:

Market need 1: need for increased efficiency. Commissioning is typically carried out during the last step of the development process. Any malfunction during this process can lead to significant time delays. Therefore, there is a need for efficient testing methods that can identify and resolve any issues from the early stages of the development process.

Market need 2: need for cost reduction. Conventional PLC testing procedures are time-consuming and expensive, given they are naturally conducted manually. In addition, PLC testing often requires physical equipment, as it is typically tested when the system is fully in place. This is costly as it can cause serious equipment damage. Therefore there is a need to cost-effectively test PLC programs without the need of any physical equipment.

Market need 3: need for improved quality. PLC programming requires considerable manual effort, which leaves the PLC open to errors due to unexpected implementation errors. PLCs are critical components in the production system, in which a simple programming error can cause severe damage, operational downtime and hazardous events. PLCs should therefore be thoroughly tested, especially if any safety-critical operation is involved.

Market need 4: need for high reconfigurability. The market demand for highly customised and reconfigurable products is increasing. However, frequent reconfigurations of PLCs make the conventional testing process even more challenging, as the system must be continuously tested every time there is a new configuration change in the layout. Therefore, there exists a need for a methodology that enables continuous integration of changes in PLCs.

Market need 5: need for automated testing methodologies. PLC testing is a time-consuming and labour-intensive process, which is predominantly performed manually. Existing manual testing practices require significant effort and time, hence the need to automate PLC testing practices.

Market need 6: need for sustainable manufacturing. The conventional mass production, and mass customisation manufacturing paradigms create value out of high-volume production, however, they are the most destructive approaches in terms of sustainability. The new manufacturing paradigm should therefore foster sustainable goals to reduce waste and maximise the reuse of existing resources to address new production processes.

With the aim of accomplishing these needs, we have presented a methodology and toolset to test PLC programs in a simulation environment. However, due to the shortcomings highlighted in Section 9.1.2, the existing approach does not address all the needs outlined above. Therefore, to be competitive in the market, we should pay particular attention to the main drivers in the following manner:

- Optimising overall testing approach execution times, particularly test execution time in TIA Portal (Market need 1).
- Automating all manual steps involved in the current approach to avoid unexpected errors, such as the conversion to PLCopen XML format (Market need 3, Market need 5).
- Migrating the digital model to a fully synchronised digital twin. The digital twin could be used to enable the continuous integration of PLCs by following a DevOps approach, as presented in the Theoretical Framework in Chapter 5 (Market need 4, Market need 6).
- Validating all IEC 61131-3 FCs and FBs in TIA Portal (applicable to all market needs).

D.2 Integrated Business Model

For the satisfactory deployment of technology disruptions in industry, new technological development must be accompanied by an application. This is achieved through the development of effective business models, which are key to identifying the value proposition, capturing the value, and creating the value for the stakeholders, as indicated in Figure D.1.

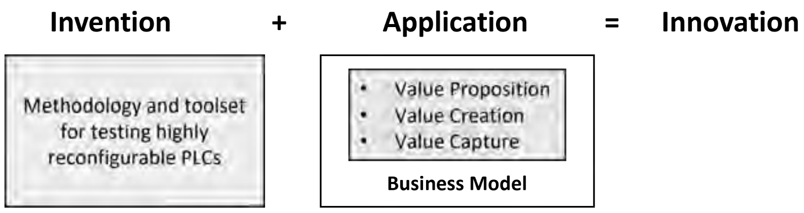


Figure D.1: From invention to innovation [MGN19]

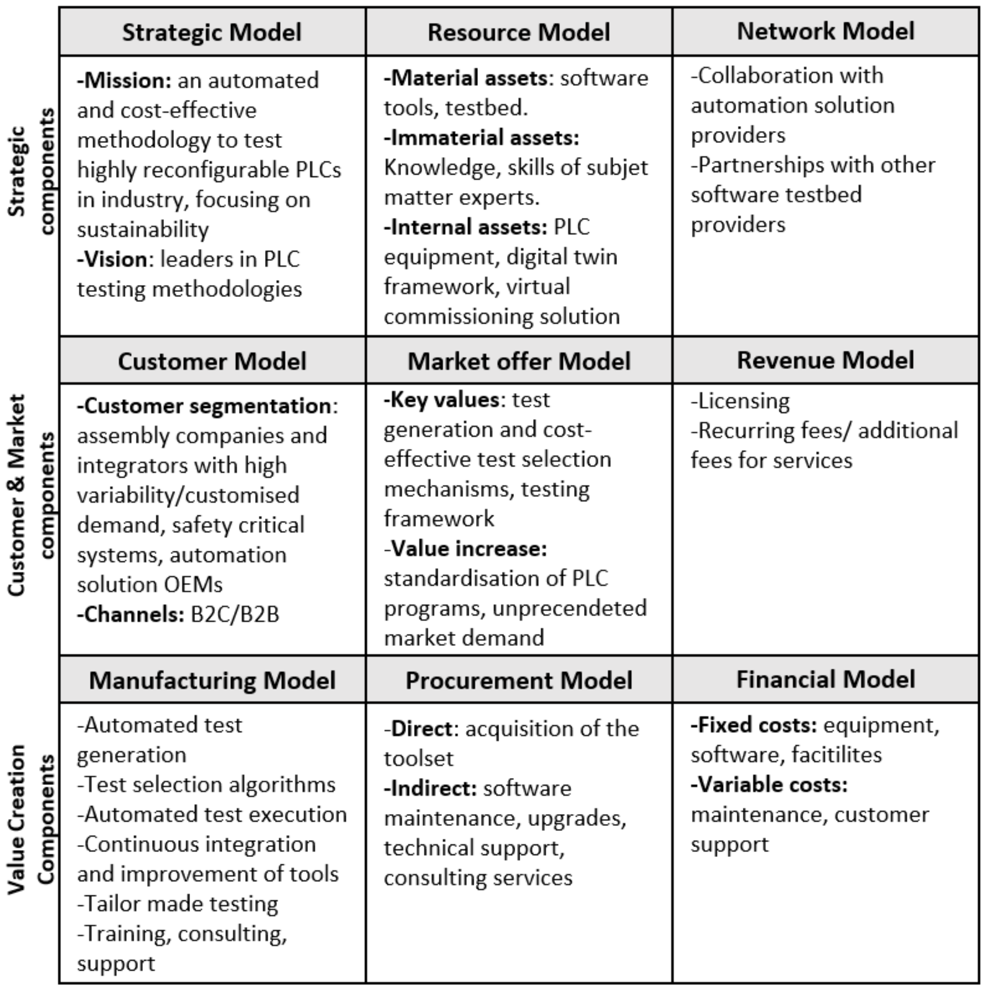
To this end, we have used the Integrated Business Model (IBM) from [WPUG16a] to define the business model of our methodology and toolset to test highly reconfigurable PLCs. It is important to note that when designing the IBM, we are considering a fully deployed solution that addresses all the specified market needs in Section D.1.

The IBM comprises three groups of components, including strategic components, customer and market components and value creation components. Each component consists of three models, resulting in a nine-component business model, as indicated in Figure D.2. These components are briefly described below:

Strategic components

- **Strategic model:** the main mission of this research is to develop an automated and cost-effective methodology for testing highly reconfigurable PLCs in the industrial sector, with a strong focus on sustainability. The aim is to reduce commissioning time and costs, increase reliability and productivity, and maximise resource reconfigurability and reutilisation. Furthermore, this methodology aids humans by reducing manual repetitive and routine tasks. The long-term vision of our strategy is to become the leading provider of PLC testing methodologies in the manufacturing sector through a spin-off or an alliance with a company.

- **Resource model:** we identified material, immaterial and internal assets. Material assets include software tools and a testbed for testing PLC programs. Immaterial assets include the knowledge, expertise, and skills of software and automation engineers, which should work in a collaborative manner. The internal assets consist of PLC equipment, a digital twin framework, and a virtual commissioning solution, including all required simulation and emulation technologies.



	Strategic Model	**Resource Model**	**Network Model**
Strategic components	**-Mission:** an automated and cost-effective methodology to test highly reconfigurable PLCs in industry, focusing on sustainability **-Vision**: leaders in PLC testing methodologies	**-Material assets**: software tools, testbed. **-Immaterial assets:** Knowledge, skills of subjet matter experts. **-Internal assets:** PLC equipment, digital twin framework, virtual commissioning solution	-Collaboration with automation solution providers -Partnerships with other software testbed providers
	Customer Model	**Market offer Model**	**Revenue Model**
Customer & Market components	**-Customer segmentation**: assembly companies and integrators with high variability/customised demand, safety critical systems, automation solution OEMs **-Channels:** B2C/B2B	**-Key values**: test generation and cost-effective test selection mechanisms, testing framework -**Value increase:** standardisation of PLC programs, unprecendeted market demand	-Licensing -Recurring fees/ additional fees for services
	Manufacturing Model	**Procurement Model**	**Financial Model**
Value Creation Components	-Automated test generation -Test selection algorithms -Automated test execution -Continuous integration and improvement of tools -Tailor made testing -Training, consulting, support	-**Direct**: acquisition of the toolset **-Indirect:** software maintenance, upgrades, technical support, consulting services	**-Fixed costs:** equipment, software, facitilites **-Variable costs:** maintenance, customer support

Figure D.2: Integrated Business Model components of our methodology – based on [WPUG16b]

- **Network model:** a close collaboration with leading automation solution providers (e.g. Siemens) is key to promoting and distributing our testing methodology to a wider customer segment. In addition, we should establish partnerships with other software testbed providers to stay up to date with the latest testing tools and technologies for commissioning PLCs.

Customer and market components

- **Customer model:** we identified four key customer segments. The first segment comprises assembly companies or systems integrators that use PLCs in their daily operations and have high demand variability, such as the automotive sector. The second segment focuses on companies that use PLCs in their operations and work with safety-critical systems, such as nuclear plants. The third segment includes assembly companies or systems integrators that use PLCs in their operations but have customised market demand, such as the aerospace industry. Lastly, our customer segment also encompasses automation equipment manufacturers and PLC vendors such as Siemens.

We aim to provide customised support and testing solutions, based on the needs of our customer segment. To this end, our customer model focuses on two main customer channels, Business to Customer and Business to Business. The former offers direct sales to system integrators and assembly companies, whereas the latter establishes a partnership with automation equipment manufacturers and PLC vendors.

- **Market model:** our key values are the capability to generate automated test cases for FBD programs, cost-effective testing of PLC programs, and a testing framework for testing PLCopen standardised IEC 61131-3 FBD programs. These value offerings differentiate us from other companies testing solutions, which are mainly focused on functional testing (e.g. the TwinCAT unit testing framework). These values will be further strengthened if new regulations enforce the standardisation and interoperability of PLC programs. In addition, volatile markets with unprecedented market demand in terms of volume, product variability and customised requirements, will favour the continuous integration of PLC programs.
- **Revenue model:** our main revenue stream could be licensing, where we offer our toolset in exchange for licensing fees. In addition, we could have additional revenue streams through recurring fees or additional fees for providing services such as maintenance, training, consulting, and support.

Value creation components

- **Manufacturing:** key activities involve automated generation of test cases based on IEC 61131-3 standard FBD programs, development of cost-effective test selection algorithms and metrics, automated execution of test cases, continuous integration of PLC programs, continuous improvement of the toolset, tailor-made testing solutions, and additional services such as training, consulting, and support.
- **Procurement model:** this includes the combination of direct and indirect procurement models. Direct procurement involves the acquisition of the software toolset for testing PLC programs in commercially available automation solutions. In contrast, indirect procurement includes software maintenance and upgrades, providing technical support and consulting services.

- **Financial model:** the cost structure incurs fixed costs such as equipment costs, software costs and facilities, and variable costs such as maintenance costs and customer support. However, our approach offers the end users several benefits and cost advantages compared to traditional testing procedures. On the one hand, our cost-effective testing approach results in cost savings associated with time to market. On the other hand, we offer higher reliability, which reduces costs associated with implementation errors.