

**A HOLISTIC MODELING AND SIMULATION FRAMEWORK OF
COMPLEX PRODUCT DEVELOPMENT TO SUPPORT
ENGINEERING MANAGEMENT**

A Thesis
Presented to
The Academic Faculty

By

Sean Ryan Mueller

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in the
George W. Woodruff School of Mechanical Engineering

Georgia Institute of Technology

August 2025

© Sean Ryan Mueller 2025

**A HOLISTIC MODELING AND SIMULATION FRAMEWORK OF
COMPLEX PRODUCT DEVELOPMENT TO SUPPORT
ENGINEERING MANAGEMENT**

Thesis committee:

Dr. Dimitri N. Mavris, Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Oliver Sawodny
Institute for System Dynamics
University of Stuttgart

Dr. Andrei G. Fedorov, Co-Advisor
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Christina Tarín
Institute for System Dynamics
University of Stuttgart

Dr. Olivia J. Pinon Fischer
School of Aerospace Engineering
Georgia Institute of Technology

Date approved: May 02, 2025

The greatest danger in times of turbulence is not the turbulence.

It is to act with yesterday's logic.

Peter F. Drucker

To my parents, for their endless support and encouragement.

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to Dr. Dimitri Mavris for his invaluable guidance throughout my academic journey. His mentorship and the opportunity to be part of the Aerospace Systems Design Laboratory (ASDL) have had a profound impact on my academic and professional development. ASDL exposed me to a wide range of compelling projects and allowed me to develop new skills that have opened doors for my future career.

I am especially thankful to Dr. Olivia Pinon Fischer for her dedicated support and insightful advice throughout the course of this process. Her guidance in defining the topic, helping to structure the work, and thoughtful feedback were crucial to the quality of this thesis and are greatly appreciated.

I also wish to thank Dr. Andrei Fedorov, Dr. Oliver Sawodny, and Dr. Christina Tarín for serving on my thesis committee. Their involvement in the joint degree program between the University of Stuttgart and the Georgia Institute of Technology, which I was fortunate to be selected for, has made a valuable contribution to my academic journey.

I extend my appreciation to the teams at ASDL and Capgemini involved in the research initiative from which this thesis emerged. Their meaningful insights, perspectives, and discussions helped shape this work. I hope that my research contributes to and supports the continuation of that larger effort.

I am also grateful to my friends and family for their unwavering support and kindness. Their thoughtful words and steady belief in me have meant the world to me throughout this journey.

Finally, I want to thank my loving partner, Lea, for her patience and support — especially across the distance. Her understanding and encouragement have been a constant source of strength and motivation.

TABLE OF CONTENTS

| | |
|--|--------|
| Acknowledgments | v |
| List of Tables | xi |
| List of Figures | xiii |
| List of Acronyms | xv |
| Summary | .xviii |
| Chapter 1: Introduction and Motivation | 1 |
| 1.1 Challenges of Modern Product Development | 1 |
| 1.2 Digital Tools and Technologies | 4 |
| 1.3 Research Goal | 5 |
| 1.4 Thesis Structure | 6 |
| Chapter 2: Characteristics of Product Development | 8 |
| 2.1 Domains of Product Development | 9 |
| 2.2 Complex Products | 10 |
| 2.3 Product Development Processes | 11 |
| 2.3.1 High-Level Product Development Processes and Methodologies | 12 |
| 2.3.2 Interactions of individual Activities | 14 |

| | | |
|--|--|-----------|
| 2.4 | Product Development Organizations | 15 |
| 2.4.1 | Organizational Structures and Communication | 15 |
| 2.4.2 | Knowledge | 18 |
| 2.5 | Digital Tools in Product Development | 19 |
| 2.5.1 | Engineering Capabilities | 20 |
| 2.5.2 | Enhanced Collaboration and Knowledge Management | 22 |
| 2.6 | Goals and Objectives of Product Development | 24 |
| 2.6.1 | Management of Product Development | 25 |
| 2.6.2 | Challenges of Managing Product Development | 26 |
| 2.7 | Summary of Observations | 27 |
| Chapter 3: Problem Framing and Definition | | 29 |
| 3.1 | Opportunity for Modeling and Simulation | 29 |
| 3.1.1 | Benefits of M&S of Product Development | 29 |
| 3.1.2 | Potential of the Digital Twin of an Organization | 30 |
| 3.2 | Research Objective | 31 |
| 3.3 | Modeling Purpose | 32 |
| 3.3.1 | Overall PD Performance — Measures of Effectiveness | 33 |
| 3.3.2 | Insights to Motivate Changes — Measures of Performance | 34 |
| 3.4 | Modeling Content | 38 |
| 3.5 | Scope of the Framework | 39 |
| Chapter 4: Modeling and Simulation of Product Development | | 42 |
| 4.1 | Overview of Modeling and Simulation Methods | 42 |

| | | |
|--|--|-----------|
| 4.2 | Activity Networks — Discrete Event Simulation | 43 |
| 4.3 | Rework Loop — System Dynamics | 45 |
| 4.4 | Agent-Based Simulation Approaches | 46 |
| 4.4.1 | Design Cognition and Strategies | 46 |
| 4.4.2 | Team Interaction | 47 |
| 4.5 | Evaluation of the M&S Methods | 48 |
| 4.6 | Research Gaps | 50 |
| Chapter 5: Problem Formulation | | 52 |
| 5.1 | Baseline Model Selection | 52 |
| 5.2 | Extension of the Process Logic | 53 |
| 5.2.1 | Limitation of the Baseline Model | 53 |
| 5.2.2 | Applying a Problem-solving Cycle to the Product Architecture | 55 |
| 5.3 | Modeling of Digital Tools | 56 |
| 5.3.1 | Digital Tools as enabling Resources | 57 |
| 5.3.2 | Digital Tools as Platforms for Knowledge | 58 |
| 5.4 | Summary of the Problem Formulation | 59 |
| Chapter 6: Proposed Framework and Methodology | | 61 |
| 6.1 | High-level Model Architecture | 61 |
| 6.2 | Generation of the detailed Process | 63 |
| 6.3 | Addition of Digital Tools to the Simulation | 65 |
| 6.4 | Notional Case Study for Testing the Framework | 67 |

| | |
|---|------------|
| Chapter 7: Implementation of the M&S Framework | 70 |
| 7.1 Product Architecture | 70 |
| 7.2 Detailed Simulation Logic | 72 |
| 7.2.1 Technical Work | 73 |
| 7.2.2 Processing Information | 81 |
| 7.2.3 Collaboration and Consultation | 84 |
| 7.3 Simulation Execution | 88 |
| 7.3.1 Data Extraction | 89 |
| 7.3.2 Model Calibration and Verification | 92 |
| | |
| Chapter 8: Results and Analysis | 93 |
| 8.1 Analysis of Baseline Simulation Results | 93 |
| 8.1.1 Monte Carlo Simulation Results | 93 |
| 8.1.2 Dashboard for Detailed Insights | 95 |
| 8.1.3 Analysis of the Process Behavior | 98 |
| 8.2 Sensitivity Analysis | 100 |
| 8.2.1 Experiment 1 — Accuracy of Simulation Tools | 100 |
| 8.2.2 Experiment 2 — Interoperability between Tools | 103 |
| 8.2.3 Experiment 3 — New Tool Capability | 105 |
| 8.2.4 Experiment 4 — Exploration of the Organizational Design Space . . | 109 |
| 8.2.5 Summary of the Effects of Digital Tools | 114 |
| | |
| Chapter 9: Conclusion | 116 |
| 9.1 Review of the Hypotheses | 116 |

| | | |
|---|--|------------|
| 9.2 | Review of the Simulation Framework | 118 |
| 9.2.1 | Modeling Content | 118 |
| 9.2.2 | Capabilities and Limitations | 120 |
| 9.3 | Future Work | 121 |
| 9.3.1 | Model Validation and Fine Tuning | 121 |
| 9.3.2 | Framework Improvements and Extensions | 122 |
| 9.3.3 | Development of a Digital Twin of an Organization | 123 |
| Appendices | | 125 |
| Appendix A: Baseline Model | | 126 |
| Appendix B: Model Parameters | | 129 |
| Appendix C: Input Data of the Notional Case Study | | 131 |
| Appendix D: Simulation Code and Data | | 138 |
| References | | 139 |

LIST OF TABLES

| | | |
|-----|---|-----|
| 2.1 | Summary of Observations. | 28 |
| 4.1 | Evaluation of relevant M&S approaches based on the attributes of PD. . . . | 49 |
| 6.1 | Tools defined for the baseline organization and their capabilities. | 68 |
| 7.1 | Summary of the input parameters for the simulation model. | 89 |
| 7.2 | Summary of the measures of performance extracted during the simulation. . . | 90 |
| 8.1 | MOEs and MOPs of the baseline configuration. | 95 |
| 8.2 | Setup of Experiment 1. | 101 |
| 8.3 | Setup of Experiment 2. | 103 |
| 8.4 | Setup of Experiment 3. | 105 |
| 8.5 | Overview of Experiment 4. | 109 |
| 9.1 | Comparison of the developed M&S framework with the existing approaches. | 119 |
| B.1 | Interface types and possible severity values. | 129 |
| B.2 | Knowledge domains and their possible values. | 129 |
| B.3 | Tuning parameters and settings of the simulation model. | 130 |
| C.1 | Properties of the defined product elements. | 131 |

| | | |
|-----|---|-----|
| C.2 | Duration and learning rates for the high-level PD activities. | 132 |
| C.3 | Responsibilities assigned to the engineer agents. | 133 |
| C.4 | Baseline properties of the defined engineer agents. | 134 |
| C.5 | All defined tools and their capabilities. | 136 |
| C.6 | Baseline properties of the defined tools. | 137 |

LIST OF FIGURES

| | | |
|-----|---|----|
| 1.1 | Complexity and development time of some systems from 1960-2010. | 2 |
| 1.2 | Top cited benefits of tools used to enable digital engineering. | 4 |
| 1.3 | Structure of the thesis. | 7 |
| 2.1 | Relationships of the product development domains. | 9 |
| 2.2 | Generic staged product development process. | 12 |
| 2.3 | Systems Engineering V for the development of complex products. | 13 |
| 2.4 | Possible relationships between product development activities. | 14 |
| 2.5 | Common organizational structures of product development organizations. . | 16 |
| 2.6 | Modes of knowledge creation during product development. | 19 |
| 2.7 | Selection of M&S tools and technologies used during product development. | 21 |
| 3.1 | Double-loop learning through a management simulator. | 30 |
| 3.2 | Iron triangle of project management. | 33 |
| 3.3 | Use and scope of the framework for evaluating product development during planning. | 40 |
| 4.1 | Comparison of abstraction levels of different simulation methods. | 43 |
| 4.2 | Exemplary process design structure matrix of information dependencies and rework probabilities between activities. | 44 |
| 4.3 | System Dynamics rework loop. | 45 |

| | | |
|-----|---|----|
| 5.1 | Comparison of Norden’s effort model and the baseline models’ smoothed simulation results. | 54 |
| 5.2 | Generic problem-solving cycle during product development. | 55 |
| 5.3 | Overview of the problem formulation and the resulting hypotheses. | 60 |
| 6.1 | Proposed high-level M&S architecture. | 62 |
| 6.2 | High-level product development process used for the proposed framework. | 63 |
| 6.3 | Exemplary application of Norden’s effort model to multiple product development stages. | 64 |
| 6.4 | Teams and engineers defined for the baseline organization. | 68 |
| 6.5 | Hierarchical product architecture of the notional drone. | 68 |
| 7.1 | State machine diagram for the behavior of engineer agents in the simulation model. | 73 |
| 7.2 | Quality types defined for product elements. | 77 |
| 7.3 | Impact of the interface quality on the solution goodness of a integrated product element. | 79 |
| 7.4 | Impact of the tool accuracy on the measured quality. | 80 |
| 7.5 | Information exchange between agents designing interdependent elements. | 84 |
| 7.6 | Improvement of knowledge through collaboration. | 88 |
| 8.1 | Lead time and cost distributions of the baseline configuration. | 94 |
| 8.2 | Dashboard visualizing data extracted during a single simulation run of a likely run of the baseline configuration. | 97 |
| 8.3 | Applied effort over time of a likely run of the baseline configuration. | 98 |
| 8.4 | Comparison of the effort distributions generated by the developed simulation framework, the baseline model and Norden’s effort model. | 99 |

| | | |
|------|--|-----|
| 8.5 | Results of Experiment 1 — Accuracy and digital literacy. | 102 |
| 8.6 | Results of Experiment 2 — Interoperability and digital literacy. | 104 |
| 8.7 | Activity network generated by the baseline configuration. | 106 |
| 8.8 | Activity network generated by adding the new HF system simulation tool. . . | 106 |
| 8.9 | Results of Experiment 3 — Interoperability of the new tool. | 107 |
| 8.10 | Results of Experiment 3 — Accuracy and usability of the new tool. | 108 |
| 8.11 | Linear regression results of Experiment 4 — Accuracy. | 111 |
| 8.12 | Linear regression results of Experiment 4 — Interoperability. | 112 |
| C.1 | Hierarchical product architecture of the notional drone. | 131 |
| C.2 | Design structure matrix of the defined drone showing the interfaces between components and their severity. | 132 |

LIST OF ACRONYMS

| | |
|-------------|---|
| ABS | Agent-Based Simulation |
| AI | Artificial Intelligence |
| API | Application Programming Interface |
| ASoT | Authoritative Source of Truth |
| CAD | Computer-Aided Design |
| CAE | Computer-Aided Engineering |
| CDF | Cumulative Distribution Function |
| CDR | Critical Design Review |
| DE | Digital Engineering |
| DES | Discrete Event Simulation |
| DMM | Domain Mapping Matrix |
| DMU | Digital Mock-Up |
| DoE | Design of Experiments |
| DSM | Design Structure Matrix |
| DTO | Digital Twin of an Organization |
| ECAD | Electronic Computer-Aided Design |
| EKM | Engineering Knowledge Management (Tool) |
| FEA | Finite Element Analysis |
| FPY | First Pass Yield |
| HF | High-fidelity |
| HIL | Hardware-in-the-loop |
| IDE | Integrated Development Environment |

LF Low-fidelity
LHS Latin Hypercube Sampling
LT Lead Time
M&S Modeling and Simulation
MBSE Model-Based Systems Engineering
MCAD Mechanical Computer-Aided Design
MDM Multiple-Domain Matrix
MOE Measures of Effectiveness
MOP Measures of Performance
PD Product Development
PDF Probability Density Function
PDR Preliminary Design Review
PLM Product Lifecycle Management
QFD Quality Function Deployment
R&D Research and Development
SD System Dynamics
SE Systems Engineering
TPM Technical Performance Measure
V&V Verification and Validation

SUMMARY

The development of new products by organizations presents many challenges. New complex and multidisciplinary products, increased customer demands and competitive pressure, as well as agile, collaborative, and concurrent processes, make developing products increasingly complicated and challenging. This requires extensive planning and management of the organization, processes, and resource allocations to develop products efficiently. Increasingly capable digital tools are available that can address these challenges and accelerate the development of new products by facilitating collaboration, information sharing, and knowledge creation. However, their implementation also is challenging and the results are uncertain.

Traditional methods for planning product development often rely on the experience of managers, are based on assumptions, and have little to no quantitative basis, resulting in frequent cost and schedule overruns, as well as quality problems. Modeling and simulation can help manage the product development process by enabling risk-free testing and evaluation of different what-if scenarios. Some methods for this already exist in the literature but have seen limited application in industrial practice due to the lack of a holistic view of product development. In particular, the impact of digital tools on product development has rarely been considered in previous simulation approaches.

Therefore, a framework for holistic modeling and simulation of product development is proposed. An extensive analysis of existing methods led to the selection of a baseline model which was adapted to include aspects of all important product development domains (product, process, organization, tools). The simulation model is an agent-based model that models every entity of these domains as an agent allowing for various interactions among entities and the analysis of the emergent behavior. The most important addition are agents representing tools used for engineering and supporting activities. These influence the accuracy of validation activities and the way information is shared between engineers.

The framework was applied to a notional case study to analyze its behavior, perform sensitivity analyzes, and demonstrate its capabilities. The results proved that the implemented simulation logic behaves as intended and represents product development more realistically than the current methods. The impact of the agents of digital tools was studied extensively through multiple Design of Experiments (DOEs) to demonstrate their behavior and discover various interactions.

The proposed simulation framework can enable informed decision-making to improve product development performance by testing multiple configurations and identifying causes of inefficiencies. This is achieved through a comprehensive analysis of the entire design space of the organization and by providing detailed information through measures of effectiveness (MOEs) and productivity (MOPs) at multiple levels of granularity.

Validation based on empirical data and through experts is still necessary to prove the practical applicability of the framework. In the future, the framework could be a stepping-stone toward the development of Digital Twins of Organizations, by adding more detail, further improving its capabilities, and integrating real data.

CHAPTER 1

INTRODUCTION AND MOTIVATION

Today's development of products and systems is faced with a multitude of challenges, significantly affecting both the efficiency and quality of their development processes. The rise of digital tools and technologies is revolutionizing traditional processes and methods, driving a digital transformation that is changing products and the way they are developed and manufactured [1, 2]. As products become smarter and more complex, they require innovative development approaches and the collaboration of various stakeholders and interdisciplinary teams [3]. At the same time, increasing global market pressures and competition necessitate greater flexibility, cost reduction, and faster development cycles [4]. In order to effectively manage and navigate these dynamic challenges, adaptive and forward thinking strategies in modern product development are needed [5].

The following section will go over the challenges of product development (PD) in more detail. After that, the benefits of digital tools for overcoming these challenges, as well as difficulties of implementing digital tools, will be discussed. The chapter ends with a brief description of the research goal and the structure of the thesis.

1.1 Challenges of Modern Product Development

Increasing Product Complexity

Complexity in modern products is driven by their multidisciplinary nature, increased number and severity of requirements, functions, and configurations. This results in an increasing number of hardware and software components, as well as interfaces and dependencies of the components [3]. Figure 1.1 illustrates this increase in complexity over the past 60 years.

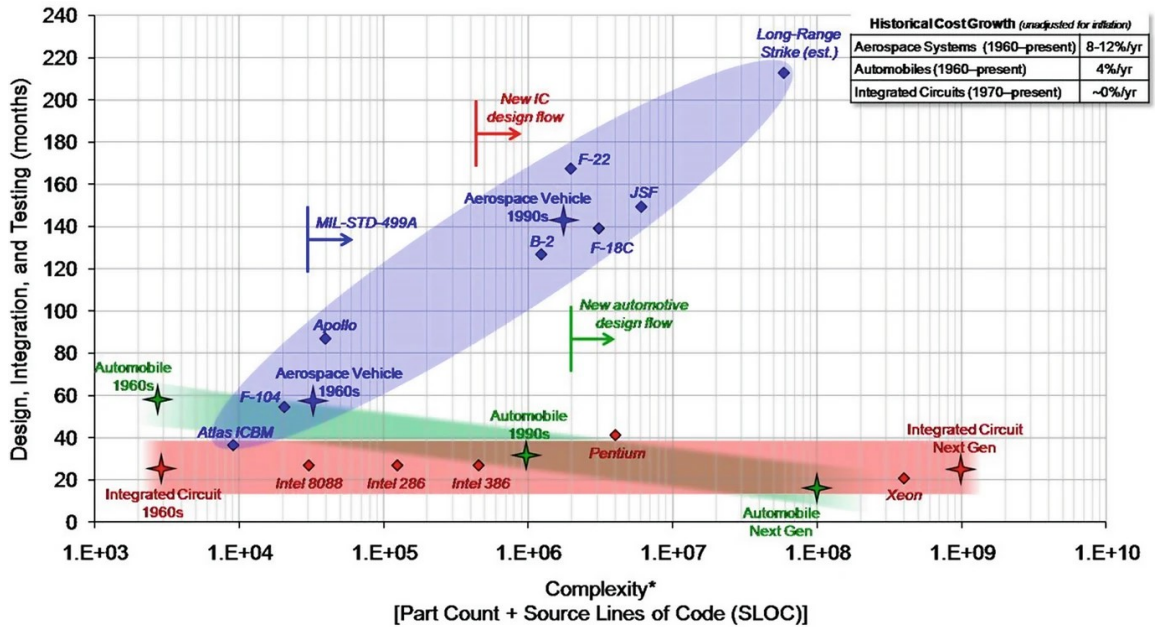


Figure 1.1: Complexity and development time of some systems over the past 60 years [6].

Products are becoming increasingly complex by integrating advanced information and communication technologies, such as the Internet of Things, Artificial Intelligence, or advanced analytics, into them [1]. This creates complex cyber-physical systems and systems of systems. These require sophisticated development processes to manage time and cost, as they involve many organizations and people working concurrently [1].

Furthermore, the trends of servicification, individualization, and reconfiguration change the development of products by increasing the involvement of customers, utilizing operational data, and requiring more flexibility and modularity in the products themselves and their development process [1]. These challenges require multidisciplinary expertise, strong collaboration among many stakeholders within multiple organizations, as well as redesigned processes and organizational structures to create successful products [1, 3].

Global Product Development

Another layer of complexity is the globalization of the development, production, and commercialization of products, leading to the need for lower cost, reduced time to market, and

the need to meet the requirements for various markets (multiple regulations and differing customer needs) [4]. This creates additional pressure on the effectiveness and efficiency of PD. The resulting global distribution of resources, including their knowledge and expertise, and complex supply chains can lead to communication and coordination barriers (e.g., geographical distance, time zones, culture, language, process compatibility) [7]. This creates further challenges for the management of PD and the efficient allocation of resources [5].

Organizational Challenges

Many of the challenges described above require changes to the organization and its development processes. These also have their own recurrent issues and challenges. The most notable issues are misaligned processes, poorly defined team structures, and problems with communication, information flow, and knowledge sharing [8]. These result in decreased efficiency or effectiveness, leading to reduced competitiveness for companies through budget overruns, missed deadlines, or reduced product quality.

These issues are especially apparent in large organizations due to the large number of teams and possible interactions [3], making cross-functional collaboration and information sharing more difficult, which are essential for the successful development of complex systems [9]. Furthermore, limited resources must be effectively allocated by maximizing their competencies and minimizing waste in unprofitable projects [5, 9].

In order to facilitate greater flexibility and resilience to changes and develop better products, alternative PD methodologies and processes (e.g., collaborative, agile, lean, or integrated) or new organizational structures are gaining popularity [10]. In addition, combined methodologies, such as integrating agile approaches into stage-gate processes, are being introduced [9]. These flexible, highly iterative, and concurrent processes make planning and predictions of the outcome of PD more difficult due to unforeseen development paths and higher degrees of collaboration and communication [10].

1.2 Digital Tools and Technologies

Digital tools and technologies can help overcome the challenges mentioned above and also lead to faster PD processes, reduced cost, and higher quality products [2, 11]. Therefore, implementing digital tools is necessary to gain or maintain competitive advantages. But their implementation also has some challenges because it requires organizational changes.

Benefits and Capabilities of Digital Tools

Advances in information and communication technology and digital tools are transforming PD by improving efficiency and enabling the development of more complex systems. These tools improve the design and analysis of products, the communication and collaboration of stakeholders, as well as the management of data and knowledge [2, 12]. This creates a well-integrated and efficient development process. Additional benefits of MBSE, which enables digital engineering, are summarized in Figure 1.2.

Classical tools such as CAD and CAE have revolutionized PD, and today's digital landscape continues to expand, offering better and new capabilities for various tasks throughout the development lifecycle [2]. Advanced simulation tools enable virtual verification and validation, leading to a reduced reliance on physical tests and early identification of issues [14]. In the context of digital engineering, digital models and data across all domains are

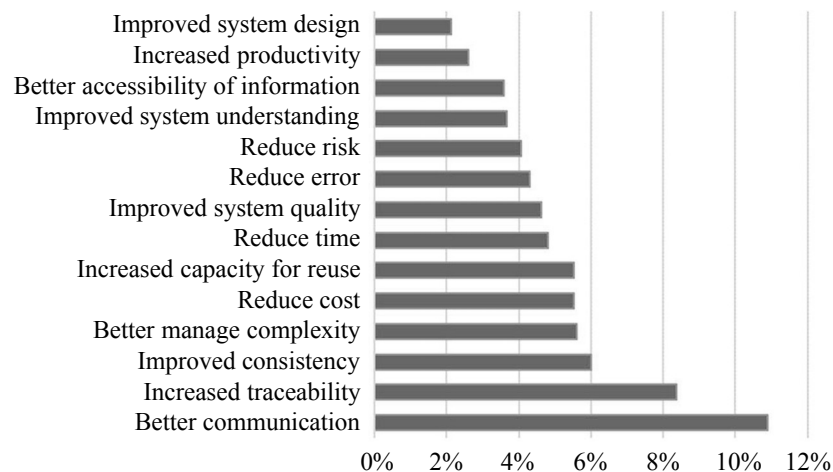


Figure 1.2: Top cited benefits of tools used to enable digital engineering [13].

integrated through MBSE and PLM tools, enabling rapid iteration and better collaboration [2, 14]. Cloud-based models and platforms allow real-time collaboration and serve as a central source of data, information, and knowledge. This improves knowledge sharing and reuse by maintaining an authoritative source of truth, enhancing consistency, traceability, and accessibility throughout the development process [15, 16].

Challenges with Implementing Digital Tools

Implementing digital tools, however, presents significant technical and organizational challenges. In addition, large financial investments are required for IT infrastructure, software licenses, and training [11, 14]. Without careful planning and consideration of the challenges, the potential benefits of digital tools can be offset by the costs and difficulties associated with their implementation [17].

One challenge is that processes, team structures, and responsibilities must be changed when implementing new tools, in order for them to align with the new capabilities provided [18, 19]. Furthermore, new competencies and therefore training are also required for personnel to be able to use the tools [12, 17]. In addition, the alignment and interoperability of different tools, especially legacy systems, must be considered, as fast data transfer is required for efficient collaboration and knowledge management [14, 19].

These changes can face cultural and personal resistance that could lead to difficulties in the adoption and effective utilization of rapidly changing technologies [11, 20]. Therefore, organizations must promote a culture of continuous learning and offer training programs to align the level of competency of personnel with the tools requirements [12, 17].

1.3 Research Goal

Overall, the challenges facing today's PD organizations are multifaceted and often require companies to strategically adapt themselves through the adoption of new tools and technologies, as well as restructuring their processes and organization. Identifying and imple-

menting the necessary changes to react to these challenges and managing PD effectively ensures that organizations can innovate and compete in the dynamic global market.

However, despite advances and research in systems engineering and project management practices, PD projects still regularly have budget and cost overruns or quality issues [21, 22]. These issues are especially prevalent in complex projects and are often even expected [23]. Although project management and planning are not the sole causes of these problems, they are an important factor [21].

Furthermore, current methods for forecasting the impact of digital tools or planning their implementation often rely on surveys and best practices (e.g., [14], [20]) or conceptual frameworks (e.g., [24], [25]). Although these provide valuable insights, they are often very broad and do not provide quantitative metrics. This makes many of these methods not applicable for direct implementation purposes, leading to the need for new methods to evaluate the impact of digital tools on PD.

Based on these observations and the general interest of academia in optimizing and reducing the risk of PD processes [26, 27], the overarching research goal emerges.

Research Goal: *Develop a method for better planning and management of product development in the context of digital engineering to efficiently identify problems as well as opportunities for improvement.*

1.4 Thesis Structure

In the chapter following this introduction, the necessary background information on complex PD is introduced. Based on the motivation and observations made about PD, the problem and objective of this thesis are further defined and scoped in Chapter 3 which leads to the opportunity for modeling and simulation (M&S). M&S methods of PD processes are then presented and analyzed in Chapter 4. The gaps of the existing M&S methods then lay the foundation for the formulation of the problem and the hypotheses of this thesis in

Chapter 5. The proposed modeling and simulation framework to address these problems and objectives is then presented in Chapter 6. The detailed realization and implementation of the framework is described in Chapter 7. The implemented framework is then analyzed and tested by performing various experiments based on a notional case study in Chapter 8. Finally, in Chapter 9 the hypotheses are reviewed, and future work is proposed. Figure 1.3 visualizes the structure of the thesis and depicts the logical flow between the chapters.

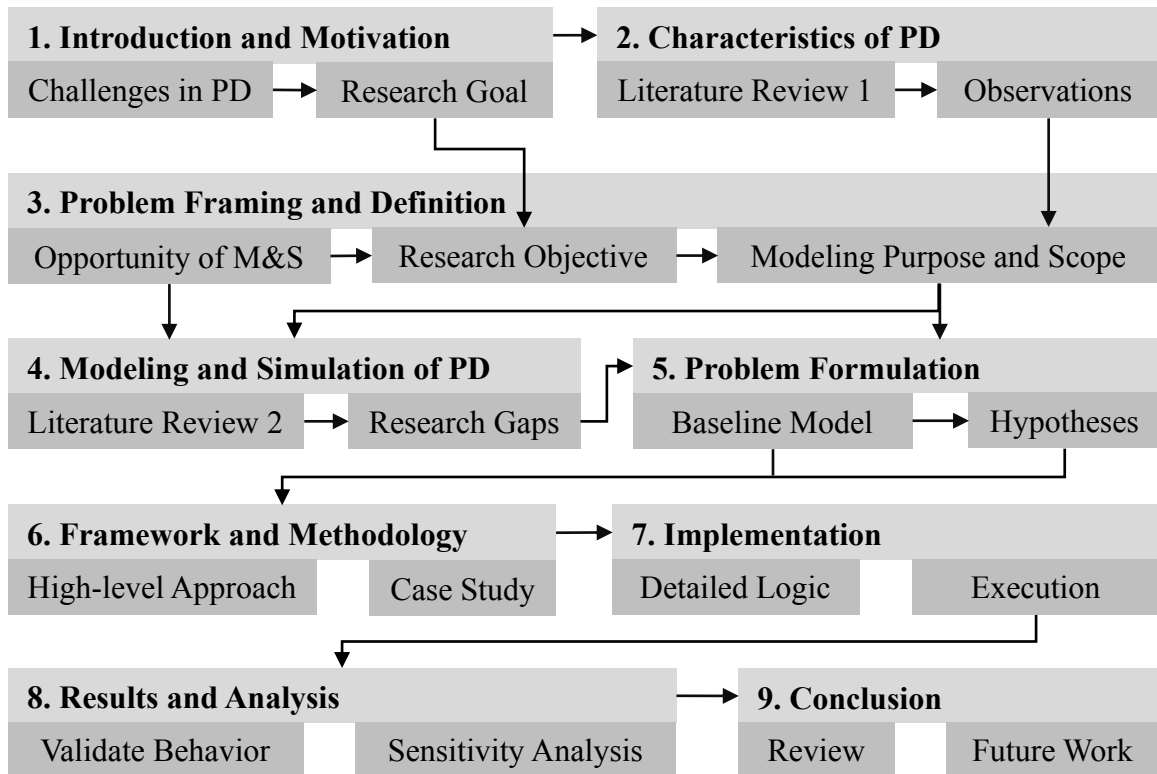


Figure 1.3: Structure of the thesis.

CHAPTER 2

CHARACTERISTICS OF PRODUCT DEVELOPMENT

Product development (PD) is defined as a

“interdisciplinary corporate process used to design a marketable product, the process is based on the definition of initial objectives and requirements for the product which are constantly further improved and iteratively adjusted in the course of the process.” [28]

The process begins with planning and the formulation of the requirements, progresses through design, development, and testing, and finally culminates in production ramp-up and product launch [29]. It integrates various disciplines and requires cross-functional collaboration to ensure that the product meets the needs of the market and customers, adheres to regulatory standards, and achieves company goals.

To better understand PD, this chapter aims to present the fundamental aspects and characteristics of product development that must be considered when creating a method for planning and managing PD. This chapter is therefore guided by the following formulation question.

Formulation Question 1: What key elements and interactions are necessary to describe and understand product development?

The following section presents an organizing framework for PD, dividing it into distinct domains. After that, the single domains will be described in depth and various observations will be derived. A short summary of these observations can be found at the end of this chapter. These observations lay the foundation for the investigation of methods for evaluating PD processes in the following chapters.

2.1 Domains of Product Development

When describing product development, scholars often refer to a few distinct domains that are used to differentiate between the different aspects and entities involved. There is no standard definition for these domains, but often categories with similar meaning are used [26]. For this thesis, an adaptation of the ZOPH¹ model developed by [30] with the following five main domains will be used [31]:

Product — outcome of product development (product design, artifacts),

Process — work and activities to create the product,

Organization — people and teams performing the activities,

Tools — software, equipment, facilities, etc. used by the people,

Goals — requirements, objectives, and policies for all of the other domains.

These domains enable a holistic description of a product development project, the entities involved, and how they relate to each other. All of these domains are additionally influenced by the environment in which the project operates. The environment includes influences of suppliers, competitors, customers, technology, research, and laws/regulations, to name a few [30]. Figure 2.1 structures the PD domains and their relationships.

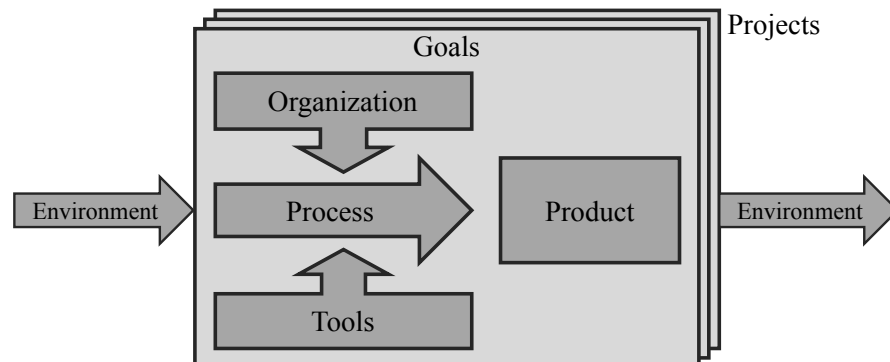


Figure 2.1: Relationships of the product development domains (adapted from [31]).

¹ZOPH is a German acronym that stands for *Zielsystem* (goal system), *Objektsystem* (object system), *Prozesssystem* (process system), and *Handlungssystem* (agent system).

The relationships of the domains can be summarized as follows: The product development process performed by the organization and its tools and resources creates a product while operating within the constraints of the environment and working towards fulfilling the strategic and operational goals [31]. The following observation can be made.

Observation 1: *Product development is made up of distinct, but interdependent domains that must be aligned with each other for its successful outcome.*

The following sections will present the most important aspects and, especially, the interactions of these PD domains.

2.2 Complex Products

The product design is the result of product development. The design describes how the product is shaped, what functions it performs, and how it is manufactured. This information is stored in a series of evolving technical artifacts such as CAD models, bills of materials (BOM), software codes, or architecture diagrams that are generated, exchanged, and changed throughout the development process [30].

A complex product or system (e.g., airplanes, automobiles) is usually made up of many multidisciplinary subsystems that are integrated with each other in order to achieve the overall system functions and requirements. These subsystems are made up of more subsystems and components that have their own functions and requirements. This decomposition of the system into subsystems and components creates the product architecture [32]. Each element in the product architecture relates to at least one technical artifact.

The complexity of the system increases not only through the large number of components and functions but also through their interactions, dependencies, or interfaces [32, 33]. These must be managed and coordinated to ensure that components and subsystems can be integrated with each other. Interfaces can have varying types (e.g., geometric/spatial, struc-

tural, energy, material, and information/data)² and severity that lead to different integration challenges and required coordination efforts [35, 36]. Furthermore, integrating systems or components from different technical domains creates additional interface complexity [37]. Therefore, it is common practice for interfaces to be defined in early development or system design, ensuring that subsystems and components can be developed independently without creating large-scale rework due to them not being compatible [6, 26].

Additionally, when developing products, subsystems and components are defined by sets of variables or parameters that make up the design space. The overall design must be optimized (multidisciplinary design optimization) with respect to the requirements by varying these design variables [33]. The complexity of this is that the optimal design for a component might not lead to the optimal system-level design [32]. Therefore, trade-offs and coordination between subsystems are necessary to find the best solution that maximizes the design objective functions related to the overall requirements of the system [6].

Overall, the complexity of the product is an important driver of the overall complexity, time, and effort of PD [3, 26, 37]. Therefore, the following observation summarizes the drivers of product complexity.

Observation 1.1: *The complexity of a product is the result of its structure (number of components, hierarchy, and interfaces), its multidisciplinary, the maturity of technologies used, and the severity of requirements.*

2.3 Product Development Processes

The process domain describes all stages, activities, and tasks, as well as their connections that are necessary for the development of a product. This section is divided into two subsections: one for the high-level overview of product development processes and one for the lower-level interactions between individual activities and tasks.

²Many other types of interfaces can be defined as described in [34]. For this thesis, the mentioned types have been selected because they are commonly used in literature.

2.3.1 High-Level Product Development Processes and Methodologies

The development of products is usually performed in phases or stages. These stages serve different purposes and are structured differently depending on the product being developed. However, the general structure of a product development process is the same for most products [29]. This idealized and generic structure of a PD process can be seen in Figure 2.2. The generic PD process encompasses several critical stages, starting with *planning*, where the project scope, goals, and resources are defined. This is followed by *concept development*, where ideas are generated, evaluated, and refined into viable product concepts. The process then progresses to *system-level design*, where the overall architecture and major subsystems of the product are defined. Next, during *detail design*, specific components, materials, and processes are developed and specified. The *testing and validation* stage follows, ensuring that the product meets all requirements and standards. Finally, during *production ramp-up*, the product is transitioned to full-scale manufacturing, where production processes are refined.

Often, instead of having strictly defined stages that have to be completed before the next stage starts (i.e., stage-gate process [38]), overlapping and iterations occur because this can greatly improve development speed and also quality [39]. In the most extreme case, development is completely iterative (i.e., agile or spiral product development). In reality, a combination of all these aspects is usually the case, and its selection strongly depends on the type of product being developed, as well as the capabilities and resources of the organization [9, 40].

More complex products require more sophisticated and concurrent processes [29]. This process is known as the Systems Engineering V [41], depicted in Figure 2.3. The system

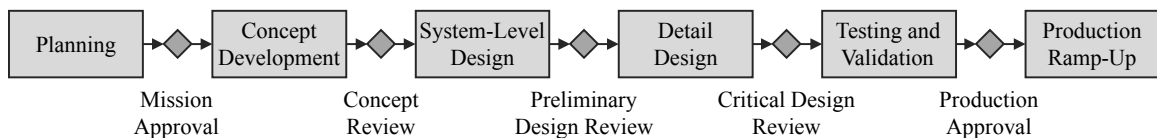


Figure 2.2: Generic staged product development process (adapted from [29]).

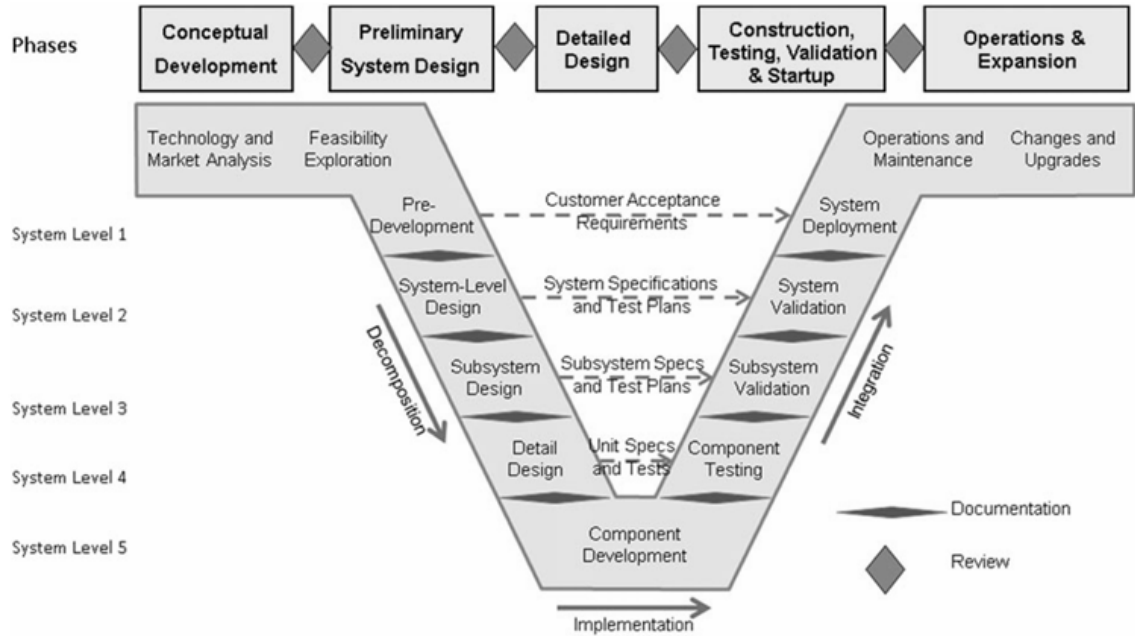


Figure 2.3: Systems Engineering V for the development of complex products [36].

design stage becomes more important for *defining and decomposing* the product in many subsystems by defining their specifications and interfaces. This makes concurrent development of the subsystems possible, which is necessary to reduce the overall development time [29]. After the development of these subsystems, they are individually tested before *integration and testing* on higher levels of decomposition occurs. During this process extensive *Verification & Validation (V&V)* of product elements is performed against the original requirements. These activities are closely related to the product architecture and its decomposition and interfaces, therefore linking the product and process domains.

Overall, the high-level structure of PD processes for complex products can be summarized as follows.

Observation 1.2: *The development of complex products follows a well-defined high-level process that guides the decomposition of the system into smaller subsystems that are then concurrently developed, tested, and integrated.*

2.3.2 Interactions of individual Activities

Stages are made up of several activities that are performed to achieve the overall result of that stage. An activity is defined as a “logically enclosed operation” [42], that needs certain inputs and generates certain outputs. Sequencing activities in a good way is important for efficient PD [26]. The main consideration that goes into this is the information dependencies between activities. Figure 2.4 illustrates what relationships can exist between activities.

Coupled information dependencies cause some activities to have to be performed with incomplete or preliminary information making the use of assumptions necessary [26]. This leads to iterations of activities when information changes or wrong assumptions were made. Finding the optimal sequence of activities to minimize rework and iteration is NP-hard [26]. However, minimal iteration will most likely not lead to minimal development time because opportunities for concurrency are not exploited. In addition, iteration can also be beneficial in increasing product quality by using newly gained insights to improve a design or correct defects [43]. Therefore, often heuristics and work policies are used to sequence PD activities.

To reduce development time, activities are often performed concurrently. The overlap of dependent activities can have a positive effect on development time, but also risks greater amounts of rework because incomplete information is used and assumptions are made [26]. In addition, more coordination is required for overlapped activities because new informa-

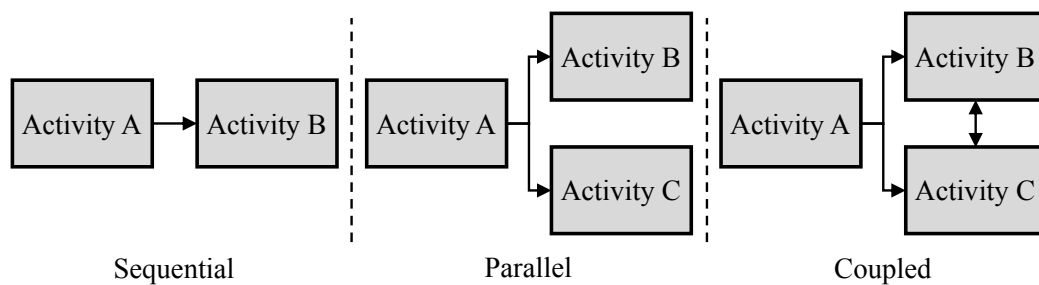


Figure 2.4: Possible relationships between PD activities.

tion is continuously generated and could cause rework [26]. These effects are described as information evolution and sensitivity. Information evolution describes how the maturity of the input information changes over time and the sensitivity describes how severe an input information change is on an activity [44]. Various work policies for overlapping and activity prioritization have been explored in the literature [44, 45]. The effectiveness of different policies greatly depends on the previously mentioned dependencies between activities.

The following observation can be made to describe the interactions between individual activities during PD.

Observation 1.3: *Product development processes are made up of concurrent and interdependent activities that necessitate the use of assumptions or preliminary information, thus resulting in iteration and rework of varying severity depending on the selected work policy and the information dependencies.*

2.4 Product Development Organizations

The organizational domain of PD describes the people in the PD organization, as well as their interactions and relationships. These people are distributed over many different departments or even organizations. Ensuring people are organized and interact with each other efficiently is therefore very important for successful product development [46].

2.4.1 Organizational Structures and Communication

Organizations can be structured in many different ways, and the chosen structure depends on many factors, such as the number of employees, the size of the portfolio, or the variety of the portfolio, to name a few [29]. The organizational structure defines who is responsible for what and how communication and interactions occur between individuals and teams. The structure influences how efficient cross-functional and cross-project communication is.

The most important organizational structures for product development are functional,

project and matrix organizations (Figure 2.5). Each of these organizational structures offers unique strengths and challenges [29, 46].

Functional organizations benefit from deep expertise and efficiency within specific departments, but can suffer from communication silos that lead to slow coordination and bureaucratic interactions. *Project organizations*, on the other hand, provide strong cross-functional collaboration and development speed, but may have difficulty building deep technical expertise and sharing knowledge between different projects. *Matrix organizations*, a hybrid structure, combine the advantages to some extent but require additional managers, leading to complex reporting and potential conflicts. Variations of the matrix organization (i.e., light weight or heavy weight team structures) are most common in well established companies of complex products because they require both highly specialized resources and strong coordination between functions [46].

Other aspects of the organizational structure are the creation of divisions (for markets, product types or geographical locations), the number of hierarchies (i.e., decision-making authority), team sizes, and the co-location or distribution of team members [46]. All of these aspects also influence the efficiency and patterns of information flow and knowledge sharing in an organization.

To develop a product, many interdependent activities are performed and coordination

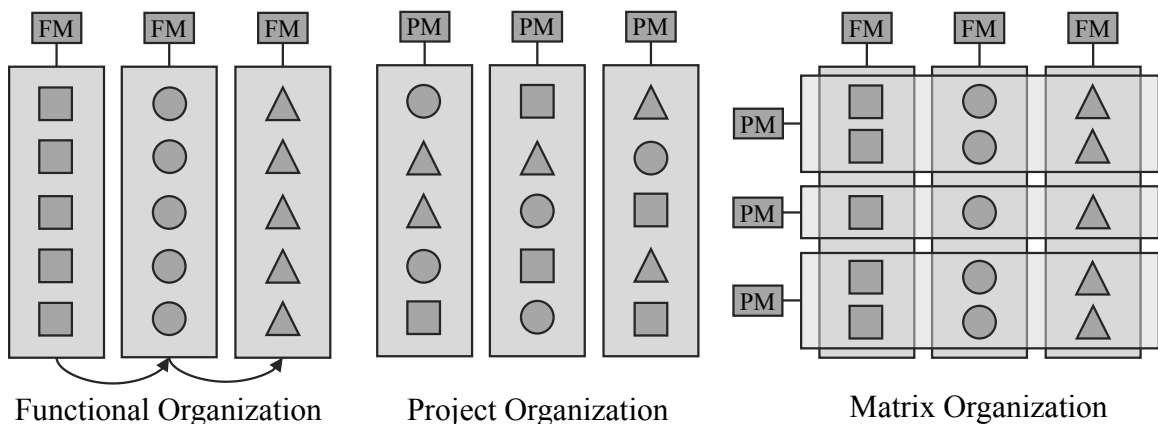


Figure 2.5: Common organizational structures of PD organizations (adapted from [29]). FM: Functional Manager; PM: Project Manager.

between subsystems must occur, which means that information flow between people, departments, or teams is necessary. An important aspect of these interactions is that the dependencies of a product mirror the communication structures of the organization that designs the product [26]. This is due to the fact that interfaces require coordination and collaboration in order to be successfully integrated. This so-called mirroring hypothesis³, also known as *Conway's Law* [50], additionally states that before development of a product can start, partitioning of its architecture into smaller subsystems has to occur and interfaces have to be identified and defined, underscoring the importance of system design. Furthermore, it was found that the amount of coordination occurring strongly depends on the development stage. During system design and integration, wide-scale coordination (overall system) is predominant, while detail design focuses more on small-scale coordination (local clusters) [51]. If interfaces are not matched by coordination efforts by the organization, the efficiency of PD is reduced because problems cannot be identified and resolved early, leading to quality issues or more rework in later stages of product development [47].

Information flow occurs through different forms that strongly influence its effectiveness and efficiency. To characterize information flow differentiation between the level of detail (e.g., documents vs. models), frequency, the medium used (e.g., email, database, telephone, face-to-face), direction (i.e., one-way, two-way), and timing (i.e., partial or complete information) are some aspects to consider [26].

The following observation about the organizational domain can be made to summarize the most important aspects of organizational structures and their communication.

Observation 1.4: *The information flow and communication, with their efficiency strongly influenced by the organizational structure, the form of information exchange used, and the active PD stage, must match the interfaces and dependencies of the product architecture in order to develop the envisioned product.*

³Many studies (e.g., [47], [48]) have mostly, with some exceptions for special and rather disruptive cases, confirmed this hypothesis but have also shown that usually not all interfaces in a product architecture are matched by interactions in the organization, leading to inefficiencies and problems [26, 49].

2.4.2 Knowledge

Product development is an information- and knowledge-intensive transformation process [52]. Input information, when combined with existing knowledge, generates new knowledge, which is then codified as information for use in other activities [31]. In the overall PD process, this is the transformation of requirements and constraints into a product design [52]. Thus, understanding how knowledge is created, shared, and used is important for understanding PD. As PD progresses, new knowledge is created, requiring continuous learning of personnel [53]. Therefore, knowledge is placed in the organizational domain because a large portion of an organization's knowledge is distributed among all engineers [54] and is related to organizational learning [55].

Knowledge is typically categorized into two types: tacit and explicit [56, 57]. *Tacit knowledge* is personal, experience-based, and often difficult to articulate, such as insights gained through hands-on work and training or deep expertise in a particular domain. *Explicit knowledge*, on the other hand, is more formalized, is easy to document and can be communicated through documents, models, or databases. Throughout this thesis, explicit knowledge will also be referred to as information because the information flow during PD is mostly done by sharing documents or models [31].

New knowledge is generated by using, sharing, and transforming existing knowledge. Figure 2.6 illustrates four different modes of knowledge creation. In PD existing knowledge (i.e., previous product documentation, knowledge repositories, expertise) is used to create new knowledge (i.e., new products, new insights) [52]. Engineers create new knowledge through experimentation, iterative design, and problem-solving activities [55, 56]. This knowledge is then codified into explicit forms, such as design specifications, technical reports, and lessons learned, making it accessible to others [58, 59]. Tools are used to store (e.g., PLM), represent (e.g., CAD), generate (e.g., FEA), or connect (e.g., MBSE) this knowledge, especially explicit knowledge [59].

Tacit knowledge is shared mainly through communication, collaboration, and consul-

tation [54, 56, 60]. This allows team members to leverage the experiences and insights of each other to exchange ideas and resolve conflicts [26]. Consultation between novices and experts is also an important factor for knowledge transfer, but also for the creation of new knowledge through interactive discussions [61]. Furthermore, knowledge management systems organize explicit and formalized tacit knowledge to make it available for reuse [60, 62]. This refers to both generalized knowledge (expertise — know-how, know-why) and product-specific knowledge (know-what) [55, 60].

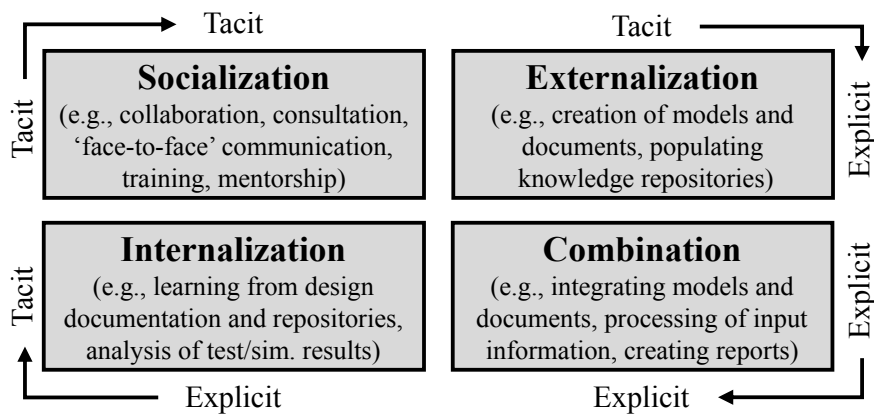


Figure 2.6: Modes of knowledge creation during product development (adapted from [56]).

Overall, the connection of the organization, product, process, and tool domains through knowledge highlights its central role in PD: The product is the embodiment of knowledge, while the process requires and transforms knowledge, which is provided and shared by people or tools [26, 52, 54, 62]. The following observation can be made.

Observation 1.5: *Product development generates knowledge (about the product) by applying existing knowledge (tacit or explicit) to activities and sharing knowledge between people and tools.*

2.5 Digital Tools in Product Development

The tool domain describes all non-human resources that are used during PD. Examples for these are facilities, testing equipment, computers, and software. These tools aid peo-

ple during tasks or are required to perform certain tasks [63]. Physical tools mainly limit product development through their cost and availability (i.e., they are mainly resource constraints). Digital tools, on the other hand, have a transformative effect on product development, changing how products are developed and tested, how people collaborate, or how knowledge and data are managed [2, 16]. This is especially the case for advanced and newer digital tools. Digital tools can speed up PD and reduce its cost, while also increasing the quality of the product [2, 12, 14]. The following observation can be made for this.

Observation 2: *Digital tools are enablers of PD that allow products to be developed more efficiently by offering new capabilities that reduce physical testing needs, promote easier collaboration, and help managing knowledge.*

Since this impact of digital tools on PD is of special interest for this thesis, the following formulation question will guide the following sections to identify in more detail how digital tools influence PD to achieve the stated benefits.

Formulation Question 1.1: *How do digital tools influence product development?*

2.5.1 Engineering Capabilities

Modern engineering relies on digital tools such as computer-aided design (CAD) and computer-aided engineering (CAE) to create, analyze, and refine products. Today, the landscape of modeling and simulation tools that are used throughout the entire PD process is continually evolving through better and new tools [2, 16]. Some of these tools and technologies used can be seen in Figure 2.7.

The constant improvement of digital tools, with increased ease of use, added features, and enhanced capabilities, ensures that engineers can create better designs and analyze them with increased accuracy, while requiring less effort [2, 64]. These effects lead to an overall improvement in the efficiency and effectiveness of PD. This is achieved through the automation of tasks, the integration of tools, and advanced tools for virtual validation.

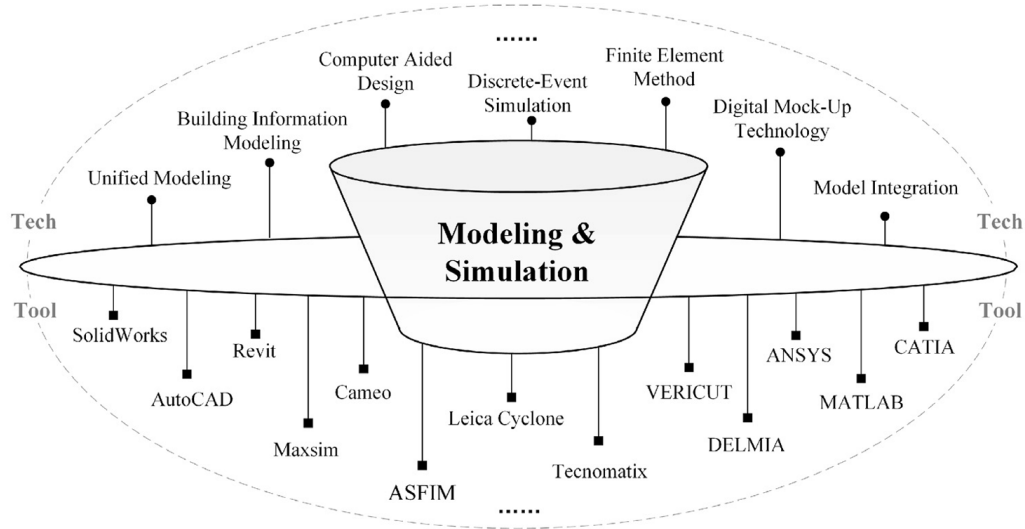


Figure 2.7: Selection of M&S tools and technologies used during PD [16].

New features and capabilities such as knowledge-based design, AI-driven assistants, task automation, and generative design methods allow engineers to quickly explore a wide range of design alternatives and reduce the time required for manual adjustments [2, 16, 64]. These are especially useful for repetitive and computationally intensive tasks. Furthermore, engineering intelligence and data analytics also lead to data-driven insights that support more informed decision making throughout the product lifecycle [16, 64].

Integrating simulation environments with each other ensures that multiple aspects of a system (i.e., mechanical, thermal, fluids, and electronic) can be analyzed simultaneously, providing faster analyzes and a more comprehensive understanding of design trade-offs [2]. Analysis capabilities within design tools also enable faster iteration between designs, reducing the dependence on specialists and additional tasks [2].

A key advantage of digital engineering is its ability to front-load problem-solving by identifying design flaws and performance issues in the early stages of development [16, 65]. Advanced modeling and simulation tools allow engineers to test components and their interactions before prototyping and manufacturing begins. Especially high-fidelity simulation capabilities, such as digital mock-ups (DMU) or digital twins, allow early validation and fast iteration to test, refine, and optimize designs much earlier [14, 64]. These so-

called virtual prototypes therefore ensure that the product meets performance requirements without needing excessive prototyping and physical testing, which reduces costly late-stage modifications.

While the well established tools are becoming increasingly easy to use and efficient [2], newer highly sophisticated tools (i.e., model-based systems engineering, data analytics, digital twins) require new competencies and other areas of expertise. In addition to digital literacy [66], more emphasis is needed on systems thinking and multidisciplinary expertise [2, 64]. Training of the workforce to maintain or build competencies is therefore very important [17]. Overall, the following observation can be made to summarize all the above-mentioned aspects.

Observation 2.1: *Digital tools have powerful capabilities to directly support or enable engineering activities (e.g., design, simulation) that enable faster, higher quality, and more cost-effective development of products, but also require new competencies and adapted processes.*

2.5.2 Enhanced Collaboration and Knowledge Management

The previous section described how digital tools create knowledge. However, they also change how knowledge is shared, stored, and reused through the use of collaborative and knowledge management tools [2]. As stated in the previous sections, collaboration, as well as efficient use and management of knowledge, is important for successful PD, especially with complex products [5].

Classical product data management (PDM) and product lifecycle management (PLM) tools are used as central repositories of product data, making information available to other people [64]. Cloud-based models and tools allow fast or even real-time collaboration, accelerating PD by reducing the need for data transfer and enabling distributed remote work [2]. Knowledge management platforms such as Wikis and expert systems are important for providing additional knowledge and information by formalizing tacit knowledge [59].

These systems also enable reuse of existing knowledge that has been generated by previous projects, which is highly desirable and is seen as a means to reduce the PD effort and lead time [64, 65]. Digital tools such as MBSE or PLM play a crucial role in facilitating reuse by organizing, indexing, and connecting the available knowledge to be easily accessible and findable [59, 64].

Digital engineering is an emerging trend that aims to transform PD by integrating digital models and authoritative data throughout a systems lifecycle as well as all the domains and disciplines involved [15]. This transition from document-centric to model-centric approach promotes consistency, better collaboration, and knowledge management by having an authoritative source of truth (ASoT) that is easily accessible and always up-to-date [16]. Digital engineering is enabled by MBSE, digital twins and a digital thread between tools and models [16, 67].

The digital thread, enabled by MBSE, is especially important for the realization of ASoT by linking digital artifacts throughout the product lifecycle to ensure consistency, traceability, and accessibility [68]. It integrates different tools, models, simulations, and data sources with each other, allowing seamless collaboration between domains while maintaining synchronization of evolving information. In addition, the reuse of knowledge and models is promoted by a digital thread [13, 68]. A prerequisite for a fully integrated digital thread is high interoperability between tools and models [19, 68]. This can be achieved through centralized platforms and unified standards for the exchange and connection of tools, models, and data, ensuring that information can be shared and used across all necessary tools [16, 19, 69].

Overall, the various mentioned tools reduce the time required to search for information, allowing engineers to focus more on engineering tasks [67]. The high availability and traceability of up-to-date information enables informed decision making, faster and early design verification, as well as easy collaboration between different technical domains and fast access to required information [13, 16, 68]. This increases productivity and reduces the

overall PD cycle time, while also reducing the number of defects and generating higher-quality products [13]. In addition, the total cost of PD is reduced, as well as the cost and effort of changes or rework [13, 14]. In summary, the following observation can be made.

Observation 2.2: *Digital tools improve the availability, organization, and flow of knowledge and information, ensuring consistency and efficient collaboration through better accessibility, traceability, and exchange between tools and personnel.*

2.6 Goals and Objectives of Product Development

The goal domain of PD describes the objectives and requirements that it should be achieved [30]. This is done by defining an adequate strategy. These goals can be defined as product requirements, such as consumer needs and regulations, or desired product performance and quality. But also process requirements are defined that set budget and schedule objectives. In addition, constraints are defined for resources on a project, such as how many people and what tools can be used or are available.

This makes the goal domain a trade-off between conflicting objectives, which constrains the PD project [31]. Furthermore, PD is inherently uncertain, making these trade-offs and planning difficult with the knowledge of the final outcome being limited [70]. Due to this, thorough planning and management of the PD domains with respect to the objectives set is necessary to make good decisions and develop a robust plan.

An organization usually has multiple projects, a portfolio of more or less different products, that are executed simultaneously [29]. These projects are likely interrelated in some way, making trade-offs between projects necessary. Different projects could use the same tools and personnel, develop modular product platforms, or follow the same company-wide strategy [63]. This often leads to different projects having to be balanced with each other and more important and profitable projects being prioritized [9, 29]. All of the above-mentioned aspects lead to the following observation.

Observation 3: *Management of PD is concerned with effectively and efficiently organizing, defining, and constraining the other domains of PD to maximize the PD outcome by achieve the objectives, goals, and requirements as well as reducing uncertainty and risk.*

For this thesis, the goal domain will be seen as the reasoning and motivation of managers to plan, coordinate, and control product development. This will be further investigated in the following sections by investigating the following refined formulation question.

Formulation Question 1.2: *How are product development projects managed and planned?*

2.6.1 Management of Product Development

Management of PD can be separated into two categories: planning and control. The goal of PD *project planning* is to define the scope, sequence and schedule of activities, allocation of resources, budget, and risk plan [29, 71]. This plan is used to prescribe processes and responsibilities during the execution of the project [70]. *Project control* is concerned with evaluating the status of the project and forecasting progress based on the available data [29, 71]. This information helps to make decisions about corrective action or identify opportunities for improvement.

Several methods and tools exist to help managers make decisions about planning and managing PD. Many of these focus on the process itself, as this is the center of PD, but often exclude the other domains. The most well-known and used method is the Gantt chart. Other methods include Critical Path Method (CPM), Program Evaluation Review Technique (PERT), Graphical Evaluation Review Technique (GERT), Critical Chain Project Management (CCPM), Business Process Model and Notation (BPMN), Integrated Definition (IDEF0, IDEF3), Event-driven Process Chain (EPC), Signposting, or Value Stream Mapping, to name a few [72, 73]. Process planning methods focus on representing PD processes as discrete tasks that interact through information transfers. They are used to

model task dependencies and iteration in order to sequence PD activities within a specific organizational or design context, and sometimes also include resource allocations.

Other approaches such as enterprise architectures (i.e., Zachmann, DoDAF, UAF) [74] or multi-domain matrices⁴ [75, 76] can be used to create more holistic representations of PD [26, 72, 73]. These connect various aspects of different PD domains with each other through their dependencies and relationships to align them with each other. However, these methods often do not provide quantitative metrics.

2.6.2 Challenges of Managing Product Development

Many of the methods mentioned above used to manage PD heavily rely on the experience of managers, are often purely static and descriptive, or make many significant assumptions and simplifications by only including the process domain. [72, 77, 78]. Additionally, some of these methods are tailored towards business processes that are inherently different from PD processes. Product development is a multidisciplinary creative process to create something new, with the outcome not being defined or known in detail at the beginning [42]. This leads to more dynamic, flexible, parallelized, and iterative processes with many uncertainties [31]. Therefore, more coordination between activities and rework of completed activities is required, creating a complex network of activities with many forward and backward dependencies, as well as many dependencies with the other PD domains. Whereas business processes often have a linear process flow of mostly independent tasks with a predetermined and easily measurable outcome [42].

These reasons make modeling PD with existing tools difficult and non-ideal [31, 70, 72]. Although simplifications are necessary for modeling, oversimplifying complex PD or making wrong assumptions due to missing expertise or knowledge pose the risk of generating inaccurate predictions and therefore leads to poor planning [31]. Especially the additional challenges and complexities of modern PD worsen these effects.

⁴Multi-domain matrices (MDM) [75] combine design structure matrices (DSM) that structure entities of a single domain with domain mapping matrices (DMM) [76] that connect entities of different domains.

Overall, current tools and methods do not provide the necessary insight and information to effectively manage PD, leading managers to make difficult decisions with high degrees of uncertainty. Thus, management and planning of PD is challenging and still flawed in many organizations, leading to failed projects, budget and cost overruns, and quality problems [21]. Although new methods are being developed to contain these problems, many do not find applications in commercial tools or industrial practice [70, 72]. This underscores the need for new methods to evaluate PD projects, as stated in the motivation and goal of this thesis. More reliable and accurate quantitative tools for the planning of PD could drastically improve the management and optimization of PD [27, 72]. This leads to the following motivating gap that is strongly related to the **Research Goal** stated in the introduction.

Motivating Gap: *Current (quantitative) planning and management methods are insufficient for evaluating the performance of product development by making many assumptions and not being based on actual organizational behavior.*

2.7 Summary of Observations

This chapter was guided by formulating questions **FQ1**, **FQ1.1**, and **FQ1.2** to identify the characteristics of product development, the impact of digital tools, and how product development is managed and planned. Because the fundamentals of PD are well understood in the literature, these questions did not directly lead to any hypotheses, but rather to a set of observations that will be used throughout this thesis as the main characteristics and attributes of product development. Table 2.1 summarizes all the observations made in this chapter. The following chapter will derive the research objective and a detailed definition of the problem based on the research goal and the analysis of complex PD in this chapter.

Table 2.1: Summary of Observations.

| Observation 1 | Interdependent Domains of PD |
|-----------------------|---|
| 1.1 | The <i>complexity of a product</i> is the result of its structure (number of components, hierarchy, and interfaces), its multi-disciplinary, the maturity of technologies used, and the severity of requirements. |
| 1.2 | The development of complex products follows a <i>well-defined high level process</i> that guides the decomposition of the system into smaller subsystems that are then concurrently developed, tested, and integrated. |
| 1.3 | Product development processes are made up of <i>concurrent and interdependent activities</i> that necessitate the use of assumptions or preliminary information, thus resulting in <i>iteration and re-work</i> of varying severity depending on the selected work policy and the information dependencies. |
| 1.4 | The <i>information flow and communication</i> , with their efficiency strongly influenced by the organizational structure, the form of information exchange used, and the active PD stage, must match the interfaces and dependencies of the product architecture in order to develop the envisioned product. |
| 1.5 | Product development <i>generates knowledge</i> (about the product) by applying existing knowledge (tacit or explicit) to activities and <i>sharing knowledge</i> between people and tools. |
| Observation 2 | Digital Tools as Enablers |
| 2.1 | Digital tools have powerful capabilities to directly support or <i>enable engineering activities</i> (e.g., design, simulation) that enable faster, higher quality, and more cost-effective development of products, but also require <i>new competencies and adapted processes</i> . |
| 2.2 | Digital tools improve the availability, organization, and flow of knowledge and information, ensuring consistency and <i>efficient collaboration</i> through better accessibility, traceability, and exchange between tools and personnel. |
| Observation 3 | Management by structuring the other PD Domains |
| <i>Motivating Gap</i> | Current (quantitative) planning and management methods are <i>insufficient</i> for evaluating the performance of product development by making <i>many assumptions</i> and not being based on actual organizational behavior. |

CHAPTER 3

PROBLEM FRAMING AND DEFINITION

Based on the **Research Goal, Motivating Gap, and Observations** made in Chapter 2, this chapter aims to define, frame, and scope the problem of this thesis.

3.1 Opportunity for Modeling and Simulation

The opportunity for modeling and simulation (M&S) to quantitatively evaluate product development is apparent and has also been demonstrated by various researchers [26, 73].

3.1.1 Benefits of M&S of Product Development

M&S of PD can allow managers to test different what-if scenarios of the organizational configuration and its processes in a risk-free virtual environment prior to actual implementation, leading to a better understanding of organizational problems and more efficient data-driven decision making [26]. By predicting the overall performance and identifying bottlenecks and inefficiencies for a given configuration, managers can obtain valuable information on how to optimize the development process. Among other things, this can help with resource allocation, process planning, continuous improvement, identifying training and hiring needs, as well as deciding whether a project is worth pursuing or not [31, 79].

Simulation models can therefore help managers and employees improve their mental models of organizational design by quickly cycling through various configurations without having to see the results in the real world. Learning then occurs in a double-loop, as depicted in Figure 3.1, by testing assumptions and observing the outcome of different decisions and strategies in the simulation model, as well as in the real world [80]. This leads to better knowledge transfer and a more generalized understanding of organizational theory which make better strategies and decision-making possible.

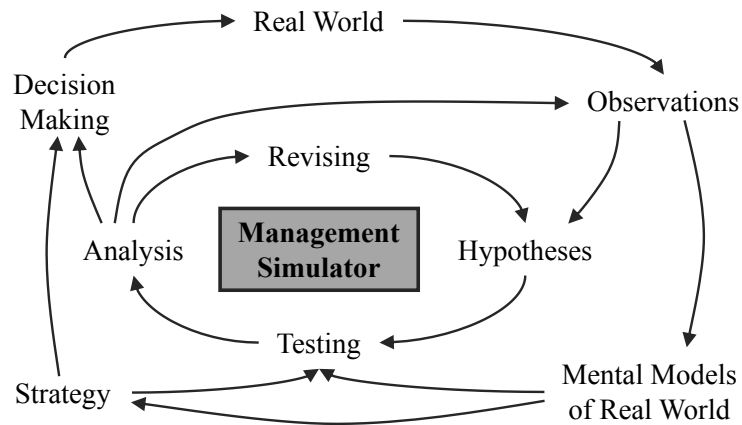


Figure 3.1: Double-loop learning through a management simulator (adapted from [80]).

Lastly, since organizational and process knowledge is often stored in managers' minds, who are not guaranteed to stay in the organization, building models of the PD process can be a way to manage and retain this knowledge and expertise [79].

3.1.2 Potential of the Digital Twin of an Organization

Further expanding on the opportunity and benefits of modeling and simulation, in recent years, the concept of the Digital Twin of an Organization (DTO), also called Enterprise Digital Twin, has gained interest among practitioners and researchers. They represent a promising, yet challenging, innovation for enhancing organizational performance through modeling, simulation, and data analytics.

A DTO, similar to Digital Twins of technical systems, is a digital representation of its physical counterpart, in this case the entire organization and all its resources, personnel, and processes, as well as their interdependencies and interactions [81]. External factors influencing an organization could also be part of a DTO. The intended purpose of DTOs is to continuously monitor and optimize the organization as a whole through the analysis of data on its state and performance using simulation models and advanced data analytics. The DTO can then identify bottlenecks, inefficiencies, and vulnerabilities, thus helping managers with decision-making, problem-solving, and the (strategic) planning of organizational processes and resource needs [82].

However, several challenges must be addressed to make DTOs viable for future applications. These include integrating diverse data sources, ensuring the accuracy and reliability of the digital model, and understanding highly complex interactions. The dynamic and non-deterministic nature of sociotechnical systems, such as organizations and their PD processes, makes simulation and predictions harder by having to account for human factors (e.g., agency, conflict, learning), hidden interdependencies of processes and structures, emergence, and continuous change [83].

To address these challenges, researchers and practitioners must develop accurate and reliable models of organizations, as well as robust methodologies for integrating and using organizational data. With data analytics (i.e. Big Data, AI, process mining) becoming increasingly sophisticated and organizational and project data becoming more available, complex behaviors and interdependencies of non-deterministic organizational processes and structures can become better understood and modeled [81, 84, 85]. Using digital traces through process mining can be used to gain retrospective insight into organizational processes [84], but foundational models are also needed to accurately describe the underlying theory of organizational behavior in order to test alternative configurations and react to previously unseen scenarios [82, 83].

Currently, DTOs are in the early stages of conceptualization and have only found limited application to PD in small case studies on the scale of individual projects (e.g. [80, 86–88]). These studies have proven the potential for DTOs to be beneficial to organizations by predicting project outcomes, evaluating changes, and helping to optimize processes and resource allocation during PD.

3.2 Research Objective

The previous section has made evident that M&S of PD and Digital Twins of an Organization (DTO) are promising concepts to drastically change and improve the way organizations and PD processes are managed. This becomes especially relevant in the context of

the increasing complexity of PD described in the introduction of this thesis.

Researchers have already developed some methods for this, but have rarely had applications in practice [70, 72] and often lack a comprehensive view of PD [26]. Therefore, more research is still necessary to identify different and potentially better ways to simulate PD [26, 27]. This is also explicitly requested and needed by industry practitioners and managers, especially in the context of virtual PD [64]. Based on this gap, the overarching research objective emerges.

Research Objective: *Create a holistic modeling and simulation framework of product development that can serve as a decision support tool for managers to identify how new configurations or the implementation of new digital tools can lead to better product development performance.*

Such a framework could serve as a baseline for the future development of quantitative methods and tools for the planning, management, evaluation, and continuous improvement of PD that project managers, R&D managers, or executives could use for organizational improvement and decision making. Although this thesis does not aim to create a DTO, the development of methods for modeling and simulating organizational components that are grounded in the theory of organizational behavior is essential for future DTO initiatives [82, 83, 86]. These can serve as foundational building blocks that, with further refinement, will pave the way for the eventual adoption of DTOs in industry.

To further define this objective, the following sections will establish the purpose and scope of the modeling framework.

3.3 Modeling Purpose

Models always serve a specific purpose and should be built appropriately for that purpose [89]. Therefore, before investigating different modeling methods, a detailed modeling purpose will be derived. Defining a purpose then limits the scope of the subsequent literature

review on available methods. This section therefore aims to derive the purpose of the modeling framework envisioned in this thesis. The basis for this are the **Research Objective** and **Observation 3** about the management of PD.

As stated in the research objective, the framework should ultimately help managers with decision-making. For this, the simulation has to provide information on the overall PD performance (Measures of Effectiveness), but also more detailed insights into the organizational behavior and performance (Measures of Performance) to motivate change. The following sections will further define these and derive the necessary modeling content.

3.3.1 Overall PD Performance — Measures of Effectiveness

PD processes are typically managed as a trade-off between cost, time, scope, and quality, also known as the iron triangle of project management (Figure 3.2) [90]. Therefore, these will be the measures of effectiveness (MOE) used in this thesis.

The scope should be understood as the product that is being developed, including all its requirements and functions. Cost and time represent the process values that are required to develop this product. Quality defines how well the scope of the product is achieved. The trade-off between product performance (scope and quality) and process performance (cost and time) [91, 92] can be seen in Figure 3.2 where the achievable scope, i.e., the area of the triangle, depends on time, cost, and quality, i.e., the length of the edges [93]. Quality itself is also dependent of the cost and time spent [91, 92]. Although these relationships oversimplify the complexities of PD [90], they provide general trends and the necessary trade-offs of the MOEs.

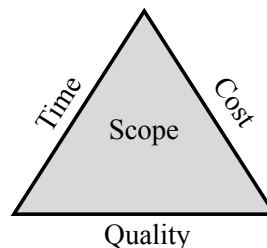


Figure 3.2: Iron triangle of project management (adapted from [93]).

Due to many uncertainties in PD it does not make sense to predict an exact value for metrics [94]. Usually, instead, probability density functions (PDF) and cumulative distribution functions (CDF) of different metrics are used [94]. These can be used to quantify PD risk, representing the potential financial impact, or value at risk, if the goals are not achieved [95]. For this, the probability of missing a goal is multiplied by its impact, represented as an impact or utility function, and integrated over all possible outcomes [95]. The utility function usually relates to market demands [95]. For this thesis, quadratic impact functions, defined in Equation 3.1 and Equation 3.2, will be used, where LT is the lead time, C is the development cost, κ_{LT} and κ_C are risk factors for scaling and unit conversion, T_{LT} and T_C are the lead time and development cost targets, and $f(LT)$ and $f(C)$ are the PDF of the lead time and cost [94]. These risk estimations can also be applied to other aspects of PD, such as technical performance measures (TPMs) [95].

$$R_{LT} = \kappa_{LT} \int_{T_{LT}}^{\infty} f(LT) \cdot (LT - T_{LT})^2 dLT \quad (3.1)$$

$$R_C = \kappa_C \int_{T_C}^{\infty} f(C) \cdot (C - T_C)^2 dC \quad (3.2)$$

The PDF and CDF can also be used to quantify robustness and reliability to quantify how resilient the process is to change and how tight the range of predictions is [96, 97].

Although the iron triangle is not perfect for fully understanding the performance and trade-offs of PD, it still provides a good overview of high-level metrics that can be augmented with more detailed and insightful metrics [90]. Some possible measures of performance (MOP) for this will be provided in the next section.

3.3.2 Insights to Motivate Changes — Measures of Performance

Measurements of the overall PD effectiveness (i.e., cost, time, quality) alone cannot motivate change, especially in the tactical and operational management layers, because there is no information on how performance can be increased. In addition, managers want to

play with different aspects of an organization that are under their control to explore different configurations and what-if scenarios [70]. These can be related to the organization, process, or tool domain of PD with some possible changes stated below [26, 98, 99]:

- *Organization*: structure, responsibilities, resources allocation, training, hiring
- *Process*: sequencing; policies for work, information exchange and decision-making
- *Tools*: availability, new tool implementation, restructuring/overhaul, customization

Identifying inefficiencies and bottlenecks or quantifying the performance of individual elements of the organization can help motivate changes that can make the greatest performance gains. This can be done through a variety of measures of performance (MOP). Some exemplary metrics will be presented next.

Breakdown of high-level Metrics

A straightforward approach to identifying key areas of interest is to break down high-level metrics into more granular components, such as specific activities, product elements, or organizational elements [100], e.g., [101–103]. This decomposition helps reveal which aspects of PD contribute most to cost, schedule and quality issues, thus motivating further, more focused analyzes. The cost of quality could also be a good metric to analyze the relationship of these high-level metrics [14].

Value-adding Work

A large portion of the work done during PD is not actual technical work. Coordination and collaboration are very important for the exchange and resolution of problems. The time used for searching for information, waiting, transportation, i.e., information preparation/sharing, or interruptions, i.e., problems, can also be attributed to PD activities [42, 68]. These supporting activities are not directly value-adding to the outcome of PD and therefore reducing these is a good way to make PD more efficient [42]. Therefore, a breakdown of all the work that is done during PD, sorted by activities or personnel, can be helpful to

identify inefficient activities. This can be expressed as the efficiency of work, that is, the fraction of the value-adding work effort to the total work effort [103, 104].

$$\eta_{value} = \frac{E_{value}}{E_{total}} \quad (3.3)$$

This metric can be especially interesting for evaluating the use of digital tools, as they can greatly impact the time required to handle (i.e., search, prepare, share) information [14, 68]. Identifying tools that cause low work efficiency could, therefore, motivate the improvement or replacement of tools.

Efficiency and Effectiveness of the Process

When planning a PD process, the schedule is of great interest to identify the staffing and tool availability requirements [29, 71], therefore, the timing of activities is a necessary insight. For this, Gantt charts are typically used. To account for uncertainty, probabilistic Gantt charts have been used for PD simulation [105], showing when activities are likely to be executed and reworked.

Plots of the work backlog [102] or the applied effort [105, 106] can be used to identify bottlenecks and define how many resources are necessary at what time during the PD process. This can be augmented by the resource utilization [101, 103] to measure how efficiently resources are used, referring to both personnel and tools.

Furthermore, rework and iteration are necessary to increase the quality of designs when validation reveals issues [43]. Rework then fixes these issues, but also causes more cost and effort. The effectiveness of PD can be evaluated by the amount of rework. This can be achieved by either counting the repetitions of activities [101], or calculating the fraction of rework-free value-adding work effort E_{eff} to total value-adding work effort E_{value} as defined below [104, 107].

$$\eta_{rework} = \frac{E_{eff}}{E_{value}} \quad (3.4)$$

Rework caused by physical tests is particularly undesirable due to significant increases in development cost and time. Digital tools can reduce the amount of physical testing required to develop a product [14]. To evaluate this impact, the ratio of virtual testing to all testing activities as well as the cost of rework could be tracked. In addition, the first pass yield (FPY) could provide insight into how well a product is designed before physical testing starts [14]. This measures how many physical tests are successful on their first run.

Information Flow

Analyzing the flow of information can help structure the organization and select or optimize tools. A breakdown of communication between personnel can help identify better team structures for different purposes [46, 108]. Capturing the information flow used for coordination, decision-making and knowledge sharing can help with analyzing an organization's capability to develop the product efficiently, but also retain and create functional knowledge can be quantified, as has been done in [7, 54, 108, 109] for example.

Analyzing the flow of information between tools can help identify which tools require better integration and interoperability [110–112]. Here, the information flow can be analyzed in terms of frequency, volume, and speed. Due to asynchronous or incomplete information transfer, some activities may not be using up-to-date information, leading to rework [113]. Therefore, measuring the consistency of the information used by different tools could reveal where unnecessary rework is caused by the use of outdated data [13].

Long-term Success

To ensure the long-term success of an organization, the evaluation of the project-specific metrics mentioned above is less relevant [90]. Long-term success is defined by the development of new technologies and organizational learning [55, 100]. This enables an organization to remain competitive in the long term by expanding its knowledge base, enhancing its technological capabilities, and thus being able to react to changing market demands. Therefore, the amount of new knowledge and competencies generated could be measured.

3.4 Modeling Content

The preceding sections have implications on what has to be modeled for the simulation. Especially the motivation to gain detailed insights into the PD process requires a high-fidelity model that represents all aspects of the organization that are of interest to managers and also are influence-able by them. This includes all relevant entities of all PD domains: product, process, organization, and tools. Therefore, the PD attributes that have been stated in **Observation 1** and **Observation 2** must be part of the M&S framework.

This also enables making less assumptions about the influence of different domains on each other as the interactions are directly modeled. This is essential to fully understand PD and its interdependencies [26], which is a problem with current methods for evaluating PD [72, 78]. To make detailed decisions about the lower-level design of an organization, individual entities that are also part of the actual organization must be modeled. This is also motivated by the fact that a DTO would also mirror the actual organization. To evaluate individual performance, data extraction must be possible for all of these entities.

The remaining PD domain, the goal domain, defines the purpose and use of the framework. As stated in **Observation 3** management of PD is concerned with organizing and defining the other PD domains. Since the impact of different configurations on performance may not always be clear, the framework must support the exploration of various what-if scenarios. This capability could help with managing PD by providing decision support through the following dimensions.

- Help planning PD by identifying the required staff, schedule, budget, and tools.
- Identification of corrective action leading to the best performance improvements.
- Test how an organization reacts to change.
- Enable the virtual part of double-loop learning (see Figure 3.1).

In summary, in addition to modeling the attributes of PD captured by the **Observations** made, the previously mentioned considerations about the modeling purpose and content lead to an additional modeling requirement.

Modeling Requirement: *The impact of the product development domains and their individual entities on each other and the overall performance has to be measured to enable detailed insights into the emergent behavior of product development.*

3.5 Scope of the Framework

In order to keep the content of this thesis manageable, the scope has to be limited. This section will define three main limitations to the scope.

Single Project

Most enterprises will have a large portfolio of parallel running projects and products that could constrain and influence each other [29, 31]. This makes planning much more complicated by requiring decisions on what projects to pursue and how to divide resources [9]. Therefore, the first limitation is that this thesis will focus on modeling a singular project. This can still be beneficial, as the understanding of modeling a single project could be adapted in the future.

Planning of Product Development

The second limitation is the focus on planning a PD project rather than executing and controlling it. The execution and control of a project requires up-to-date information on the state of the project and organization. New information or unanticipated events require the modification of the original project plan [29]. Therefore, first investigating modeling and simulation methods for planning PD is more important and can later be adapted for project execution by integrating real-time data.

Later Development Stages

The last and most important limitation is that the framework will focus on the later stages of PD. The front end of PD (deriving customer needs, requirements definition, concept development) has many uncertainties and minimal knowledge about the final outcome. Without this knowledge, planning the later stages of development (system design, detail design, testing) becomes difficult because the concept defines what subsequent tasks must be performed. Usually, a detailed plan is developed after a concept has been selected [29].

Furthermore, the design and V&V stages of PD take longer than conceptual development, which makes carefully planning these stages more important when trying to minimize the time and cost of overall PD [26, 29].

Although a well-defined concept is acknowledged to be very important for the overall technical performance and quality of a system, the developing organization also has to be able to create this concept successfully. Therefore, a framework to assess the stages of PD after the conceptual phase could be a useful tool to compare the feasibility of different concepts with respect to the capability of the organization that develops it [26].

So for this thesis, the concept, meaning the general architecture as well as functions and requirements, will be considered as given information. This enables a more detailed description and therefore simulation of the development process that should achieve the defined concept. In addition, the focus here is on the engineering aspects of PD. So aspects such as supply chain management, manufacturing, and marketing are excluded. This potential use of the framework is illustrated in Figure 3.3.

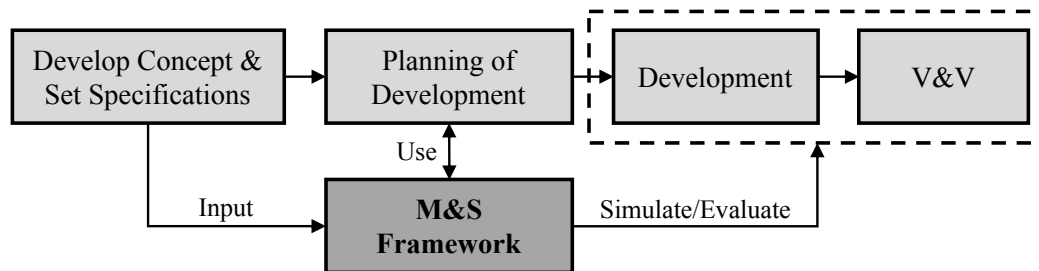


Figure 3.3: Use and scope of the framework for evaluating PD during planning.

Finally, it should be mentioned that this research is oriented towards complex engineered physical systems. The development of pure software, services, or consumable goods (e.g., clothing, food, gasoline) makes use of many different processes and methods [29]. Therefore, the framework that will be developed in this thesis would not apply to the development of these types of products.

The following chapter will review and evaluate the literature on available M&S methods of PD that could enable the research objective defined and scoped in this chapter.

CHAPTER 4

MODELING AND SIMULATION OF PRODUCT DEVELOPMENT

This chapter presents relevant related work on M&S of product development. Key concepts, modeling approaches, and methodologies will be presented in order to summarize what work has been done in the past and what their limitations are. Therefore, this chapter is guided by the following research question.

Research Question 2: *How can modeling and simulation methods be used to evaluate and plan product development?*

In the literature, there is no general consensus on what the best method is to simulate or evaluate product development, and therefore many different methods have been explored. The review in this thesis is very narrow compared to the broad available literature. The focus is on executable simulation methods, as these enable quantitative analysis of product development. For more detailed analyses of existing models and methods, refer to the literature reviews of *Yassine* [26] and *Wynn and Clarkson* [72, 73].

4.1 Overview of Modeling and Simulation Methods

In the literature on product development and especially engineering management, several simulation methods have been applied to product development with different purposes, views, assumptions, and abstractions. This section will delve into the three primary simulation methods found in the literature: *System Dynamics* (SD), *Agent-Based Simulation* (ABS) and *Discrete Event Simulation* (DES).

Each of these methods have different possible applications and abstraction levels. This is visualized in Figure 4.1. This leads to various advantages and disadvantages. A high

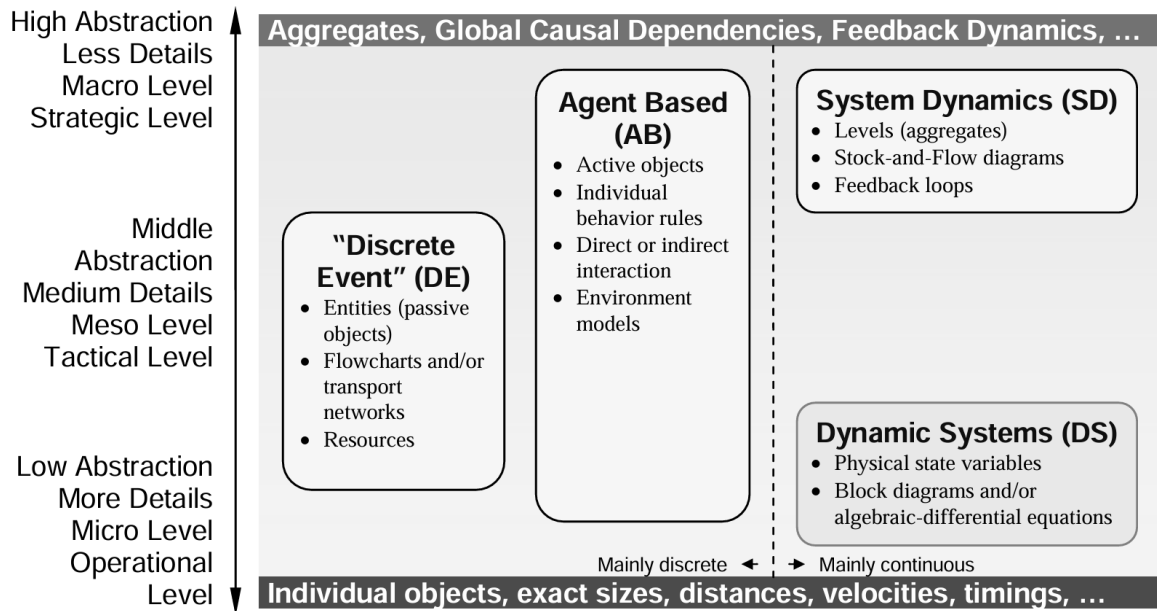


Figure 4.1: Comparison of abstraction levels of different simulation methods [114].

degree of abstraction can enable easier and faster modeling, but also reduce the level of insight, while less abstraction increases the flexibility of modeling but also the effort and complexity to create models [114].

For completeness, it should be mentioned that in addition to these simulation methods, other methods have also been proposed in the literature to quantitatively evaluate PD. The most prominent methods here include social network analysis (SNA), matrix approaches, or pure mathematical methods [26]. However, compared to the simulation methods, they do not provide as much insight, which is why they were not considered for this thesis.

The following sections will describe four commonly found types of PD simulation approaches that apply the three mentioned M&S methods. The chapter ends with a comparison and analysis of these approaches.

4.2 Activity Networks — Discrete Event Simulation

Discrete Event Simulation is a modeling approach in which the system's state changes at discrete points in time due to specific events, rather than continuously over time [114]. It

represents processes as sequences of discrete events, such as arrivals, departures, or task completions, with each event triggering state transitions.

This simulation logic has been applied to PD often and is strongly related to design structure matrix (DSM) research [26, 33, 73]. These models focus primarily on the process domain by modeling activities and their information dependencies. The DSM is used to visualize the information dependencies between activities [33]. Furthermore, the severity of information dependencies is defined by rework probabilities (see Figure 4.2) and rework impact values. These define how likely rework is due to changed inputs and how much work must be redone. Learning curve effects are usually also applied to repeating activities.

This method is commonly used to analyze the impact of different sequences of activities on rework and iteration [94, 97, 105], as well as the effects of overlapped activities and the use of different work policies [7, 44, 45]. For overlapping activities, the concepts of information evolution and sensitivity, as well as policies for information exchange, are used to define how much rework is caused by changing information during the execution of the concurrent activity. Some models also consider resource constraints [45, 97]. Generally, the flow of the process is predefined and rigid, but attempts have been made to allow more adaptive simulations by defining activity selection [115].

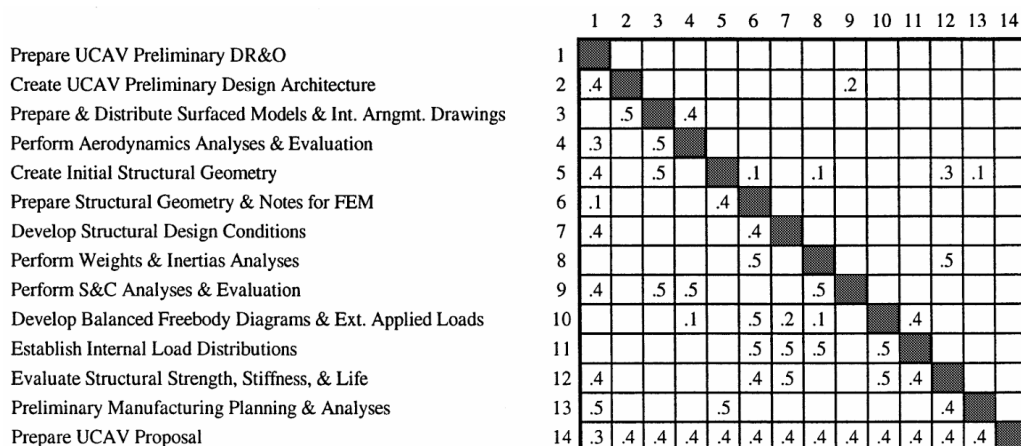


Figure 4.2: Exemplary process DSM of information dependencies and rework probabilities between activities [94].

4.3 Rework Loop — System Dynamics

System Dynamics is a simulation approach that represents systems using stocks (accumulations), flows (rates of change), and feedback loops to model how variables interact over time [114]. It focuses on aggregate-level behavior rather than individual entities, making it suitable for studying high-level trends.

For PD the “rework loop”, originally proposed by *Cooper* [116], has been used to model the iterative nature of PD from a high-level perspective. Figure 4.3 depicts the general structure of the rework loop. The following stocks are part of the standard rework loop.

- *Original Work to Do* — Planned tasks,
- *Work Done* — Work done correctly,
- *Undiscovered Rework* — Defects that have not been detected,
- *Rework to Do* — Work requiring additional effort to correct defects.

The rework loop abstracts organizational elements, aggregating them into high-level variables, often termed management levers [73]. These variables influence the flow rates (i.e., progress, error generation, rework discovery) of the model. Therefore, SD is particularly effective for analyzing high-level strategic decisions in PD. The selection of these variables can vary considerably and in extreme cases results in large causal loops between many variables related to workforce development and moral, management pressure, or the

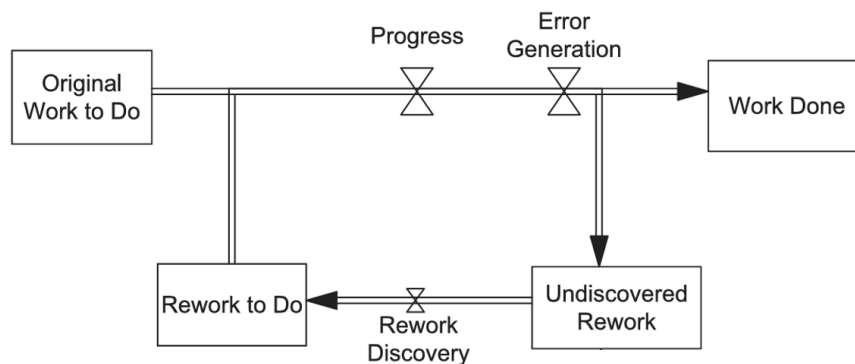


Figure 4.3: System Dynamics rework loop (adapted from [117]).

scope of the project [22, 117]. The basic rework loop can be extended by adding additional process steps or stages, each representing individual rework loops that interact with each other (e.g., [118]).

In addition to the rework loop, some authors use other approaches to model PD using SD. Examples of these applications include information maturation and ambiguity [119, 120] or smaller submodels for high-level effects (i.e., knowledge, technology, competition, cost) that are part of a DTO [80, 98]. Compared to the literature on the rework loop, these approaches are very scarce and often require detailed fine-tuning through data analytics.

4.4 Agent-Based Simulation Approaches

In agent-based simulation models consist of autonomous agents, each with individual behaviors, decision-making rules, and interactions [114]. These agents operate within an environment, responding dynamically to other agents and external conditions, leading to emergent system behavior.

For simulating PD two relevant types of ABS have been identified. These are used to model *Design Cognition and Strategies* or *Team Interactions*. The following sections will separately discuss these approaches in more detail.

4.4.1 Design Cognition and Strategies

Models of design cognition and strategies center the ABS simulation around a design space made up of interacting variables. This design space can be defined through an NK landscape⁵ (e.g., [122]), random objective functions in network models (e.g., [23]), or physics-based functions (e.g., [123]). The variables interacting in these design spaces represent interdependent technical artifacts or elements of the product being developed.

The agents associated with the individual variables change its value in an effort to op-

⁵NK landscapes are mathematical models used to study optimization problems by creating a rugged fitness landscape, where N represents the number of elements (genes, decisions) and K represents the degree of interaction between them [121].

optimize the design. Designs (i.e., values) are then exchanged between agents. Based on the interactions between variables, the individual design spaces of the agents change. This process is continued until the solution converges. Different optimization, search, communication, negotiation, or decision-making behaviors and strategies can be tested using these models [73].

4.4.2 Team Interaction

ABS models of team iterations model the product development process around agents that represent people. People can have different attributes including competencies, availability, trust, or motivation. These models are centered around simulating collaboration, coordination, and communication between individual agents. A distinction can be made between activity-based ABS models (e.g., [101–104, 124–127]) and a recently proposed knowledge-based method by *Zhang and Thomson* [106].

Activity-based Team Interaction Models

Activity-based ABS models use a process architecture, similar to the ones used in activity networks (Section 4.2), to guide simulation execution with agents working on these tasks and also interacting with each other. Depending on the responsibilities of the agents, activities are selected and worked on. This usually includes a decision logic on what activity to select based on a number of factors, such as urgency, importance, individual preference, or first/last in first out [103, 104, 125]. The flow and need for coordination and communication in these models is modeled probabilistically and defined by the information dependencies between activities and relationships between agents.

The Virtual Design Team [102, 104] is the most notable and extensively developed activity-based ABS model. It includes decision-making and exceptions that depend on organizational structures and culture, as well as information exchange and coordination between designers using communication tools or meetings.

Knowledge-based Team Interaction Model [106]

Knowledge-based ABS models of PD use a high-level process logic that is applied to a product architecture. Knowledge is used to describe the complexity of the product, as well as the competencies and experience of personnel. The process then emerges based on the complexity and dependencies of the product, as well as the competencies and knowledge of personnel. This type of model is therefore more focused on the actual need of collaboration than predefining collaboration and communication. The need is defined by quality problems that occur based on missing knowledge.

4.5 Evaluation of the M&S Methods

This chapter presented the most important methods to simulate PD found in the literature. Each of these methods serves different purposes and, therefore, have different scopes. To evaluate these methods against the purpose of this thesis, Table 4.1 assesses how well each approach satisfies the required modeling content. The modeling content is derived from the attributes of PD stated in **Observation 1**, **Observation 2**, and their sub-observations.

The product domain (1.1) is only considered in the *Design Cognition and Strategies* and *Knowledge-based Team Interaction* ABS models because they directly model the product and its complexity. Other approaches only implicitly model the product complexity by making assumptions about its impact on other domains. This can be done through information flow complexity, duration estimates, or the number of tasks to be completed.

The process domain (1.2, 1.3) is only fully considered in activity-based approaches (i.e., *Activity Networks*, *Activity-based Team Interaction*). Here, the process can be modified by using different sequences and work policies. The other models usually have a process that guides the simulation and also includes some process interactions, such as iteration, but do not use an actual process structure that could be influenced by managers.

The organizational domain (1.4) is included mainly in ABS approaches because these

Table 4.1: Evaluation of relevant M&S approaches based on the attributes of PD.

| M&S Approach | Modeling Content for a Holistic Representation | | | | | | | |
|---|--|------------------------------|-----------------------------|-------------------------------------|------------------|--------------------------|------------------------------------|----------------------------|
| | 1.1 Product Complexity | 1.2 High Level SE Process | 1.3 Process Interactions | 1.4 Organization & Communication | 1.5 Knowledge | 2.1 Tool Capabilities | 2.2 Collaboration through Tools | Req Individual Entities |
| DES — Activity Network | ○ | ● | ● | ○ | ○* | ○* | | ○* |
| SD — Rework Loop | ○ | ◐* | ○ | | ○ | | | |
| ABS — Design Cognition | ● | ○* | ○ | ● | ○ | | | ◐ |
| ABS — Team Interaction (A) | ○ | ○ | ◐ | ● | ○ | ○* | ○* | ● |
| ABS — Team Interaction (K)* | ● | ○ | ○ | ◐ | ● | | | ● |
| A: Activity-based K: Knowledge-based | | ●: Fully ◐: Partially | ○: Limited ○: Implicitly | * Only found in a few models | | | | |

models are centered around interactions, such as communication, collaboration, and coordination. Including these activities is important because much of an engineer’s daily work is related to these activities rather than the actual engineering activities [70]. The *Knowledge-based Team Interaction* models have a small limitation to this because they do not include aspects of the organizational structure. To include these interactions, ABS approaches model the individual entities of PD (Req). In particular, the *Team Interaction* models create intricate and detailed models of the entities involved.

The use of knowledge (1.5) during PD is only fully considered in the *Knowledge-based Team Interaction* models. Other approaches include some aspects of knowledge, such as competencies of personnel.

The tool domain (2.1, 2.2) has only been considered in a few simulation models compared to the amount of reviewed PD simulation literature. Some models include tools as a resource constraint (i.e., expensive software) [125, 126]. Other methods include activities that are directly linked to CAE tools, but only consider the impact of the tool implicitly

through the duration of the activity and rework probabilities [53, 128]. A different approach modeled a knowledge base that is used to resolve technical problems if no expert is available [124]. The Virtual Design Team [104] models different communication tools (e.g., meeting, email, CAD-sharing, etc.) that impact the efficiency of sharing information with other agents. In general, the impact of digital tool on PD through new capabilities and better collaboration is not adequately captured by the reviewed methods.

4.6 Research Gaps

Based on the previous evaluation, two gaps can be identified. First, it can be seen that there is no truly holistic simulation approach. Many methods focus on a specific PD domain and therefore do not or underrepresent the other domains. This leads to the first gap in the current PD simulation research landscape.

Research Gap 1: *Current M&S methods of product development often lack important aspects of the effects and interactions of the different domains by making assumptions about them or not considering them at all.*

As stated in **Observation 1**, representing only one or two domains is not sufficient to actually understand and model the intricacies of complex PD. This leads to static assumptions having to be made about the influence of the other domains or completely disregarding them, making the use of these models difficult in practical applications [72]. A holistic view is necessary because the overall performance of PD is not dependent on a single domain, but rather on the interactions and relations between these domains [26, 31]. Additionally, insights into domains that are not modeled cannot be generated by these simulation approaches, which was stated as an important aspect of the modeling purpose. It has been found that integrated methods that consider multiple aspects of PD are better suited to handle high complexity and uncertainty, as well as to better predict performance, thus leading to better planning of PD [71].

The product, process, and organizational domains of PD have been researched the most [26]. However, the tool domain has not been adequately included in any simulation model. The inclusion of digital tools in some models does not lead to significant changes in the emergent and dynamic behavior of PD because they mostly only change values instead of the actual behavior. This leads to the second gap in the current research landscape.

Research Gap 2: *Current M&S methods for evaluating product development are not able to quantify the impact of digital tools on collaboration, knowledge management, and the overall outcome of product development.*

The influence of digital tools on other domains of PD, especially knowledge management and collaboration, should be considered when evaluating them because these are important aspects of how digital tools change PD [2, 16]. Tools are often used at the intersection of the remaining PD domains to align them with each other [26]. Because tools are not considered in PD simulations, the potential benefits as well as the proper development and implementation of digital tools on the other PD domains and PD in general cannot be evaluated using them. Therefore, integrating the tools domain into PD simulations is essential to advance research in this area. [26, 27]

The identified gaps constitute the need to investigate M&S methods of PD for better approaches to integrate all domains, especially the tool domain. In the following chapter, the gaps will serve as the basis for the detailed problem formulation to address this need.

CHAPTER 5

PROBLEM FORMULATION

As stated in the problem definition, the **Research Objective** of this thesis is to: *Create a holistic modeling and simulation framework of product development that can serve as a decision support tool for managers to identify how new configurations or the implementation of new digital tools can lead to better product development performance.* The complete creation and implementation of such a framework would require a lot of effort beyond the feasible scope of this thesis. Moreover, numerous validated M&S methods already exist in the literature, as discussed in the previous chapter. Therefore, a baseline model will be selected and adjusted. For this, hypotheses will be formulated to improve this baseline model to fulfill the research objective of this thesis by filling the identified **Research Gaps**.

5.1 Baseline Model Selection

Based on the evaluation in Table 4.1 the method that best fulfills the required modeling content is the knowledge-based team interaction agent based simulation. In addition to these requirements, a few other considerations support this selection.

First, ABS models offer high flexibility [114] and therefore changes and extensions, as well as the addition of entirely new entities, are possible. Secondly, the team-interaction models are modeled around collaboration and communication between team members, and therefore already model the individual entities of PD, resulting in being able to simulate the emergent behavior of PD. Lastly, the knowledge perspective is beneficial for three reasons:

- Understanding knowledge management is of great interest in PD literature [27].
- All PD domains are connected through knowledge (**Obs. 1.5**) [54, 62].
- The integration of digital tools into the simulation is easier, because they impact the way knowledge is generated, stored, shared, and used (**Obs. 1.5, Obs. 2.2**) [2, 59].

Therefore, *Zhang and Thomson's* model [106] is selected as the baseline model for this thesis. Although this approach is fairly new and has not yet been studied extensively, it is still a promising approach because it is based on more than 20 years of research on PD [106], and it is also said to be one of the most holistic simulation models created to date, including many aspects of the product, process, and organization domain of PD [26]. In addition, the original research paper [106] describes the functionality of the model in intricate detail, which makes its usage for further research easier. A short description of this selected baseline model can be found in Appendix A.

This baseline model must be adapted to better represent the partially included characteristics in order to fill the gaps. The evaluation of the baseline model (last row in Table 4.1) determined that these are the process domain, the tool domain, and partially the organizational domain. Therefore, the following research question will guide the next sections.

Research Question 2.1: *How can the baseline model be enhanced to represent complex product development more holistically?*

5.2 Extension of the Process Logic

To identify how the process logic can be extended, the limitations of the baseline model will be elaborated first. Based on that analysis, an adapted process logic that is based on a high-level problem-solving cycle is proposed.

5.2.1 Limitation of the Baseline Model

The baseline model [106] assigns only one activity to every architectural element: a design activity for components and an integration activity for subsystems. This leads to a rigidly defined process that cannot be altered by changing the sequence or overlapping activities. Furthermore, reviews or verification activities, which check the quality of designs and decide whether iteration is necessary or not, happen instantaneously in the baseline model.

These simplifications do not accurately reflect the realities of PD, where multiple different activities may exist for each architectural element, and verification requires effort and time. As a result of this limitation, the simulation results of the baseline model indicate that most work occurs early in the PD process, i.e., many component design activities, with only a few agents remaining active in later stages, i.e., a few high-level system integration activities [106]. In development projects or individual stages, the distribution of effort usually follows a skewed normal distribution, as described in Norden’s effort model [129]. However, this characteristic is not reflected in the baseline simulation. Norden’s effort model is expressed in Equation 5.1, where γ' is the effort required (e.g., person-months per month) at time t , E is the total effort required and α is the shape parameter.

$$\gamma' = 2E\alpha t e^{-\alpha t^2} \quad (5.1)$$

Figure 5.1 compares Norden’s effort model with the approximated⁶ effort distribution of the baseline model results. It can be seen that the effort distribution of the baseline model follows an exponential distribution. Due to this misalignment of the effort curves and the highly abstracted version of the PD process, expanding the logic of the process is identified as a potential improvement of the simulation model to fill **Research Gap 1**.

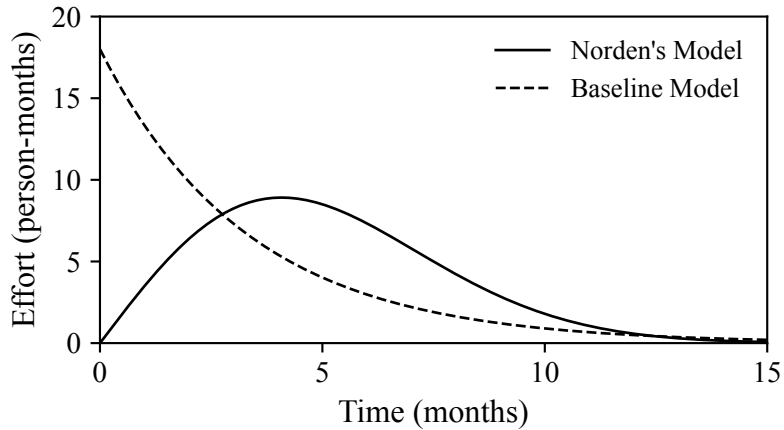


Figure 5.1: Comparison of Norden’s effort model [129], using Equation 5.1 with $E = 60$ person-months and $\alpha = 0.03$, and the baseline models’ smoothed⁶ simulation results.

⁶The results of [106] shown here have been approximated using $\gamma' = 60 \cdot 0.3e^{-0.3t}$, to reduce noise.

5.2.2 Applying a Problem-solving Cycle to the Product Architecture

As identified in **Observation 1.2** the PD process always follows a similar structure at the high level, regardless of the technical domain or product. Product development is solving a large problem. As part of product development and specifically systems engineering, this problem is decomposed into smaller problems to be solved individually and then integrated with each other later [73]. On a smaller scale, problem-solving methods are also applied iteratively until a feasible design for a subsystem or component is reached.

There are many different versions of the problem-solving cycle that can be applied to PD [73]: e.g., plan-do-check-act (PDCA), design-build-test, analysis-synthesis-evaluation, and the experimentation cycle [65]. In essence, these cycles have similar structures: (1) analyze the problem, (2) develop a solution, (3) test that solution, and then (4) evaluate the solution. Based on the evaluation, the cycle iterates, or the solution is accepted.

Figure 5.2 illustrates a generalization of the various different problem-solving cycles, inspired by a similar cycle developed by [123]. In addition to iteration inside a specific cycle, decomposition and integration have been added to combine the high-level problem-solving process with the lower-level problem-solving cycle. Decomposition or integration occurs when the individual problem-solving cycle is completed and leads to smaller or larger problems, respectively. Developed solutions on any level of decomposition and fidelity are quantified in some way (i.e., simulation or testing) to evaluate and validate them against performance criteria. The cycle continues through all decomposition and integration levels until the product is fully developed. This cycle captures the possibility of iteration not only at the component level but also at the system level [55].

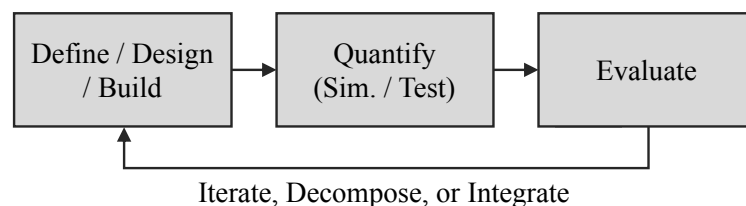


Figure 5.2: Generic problem-solving cycle during product development.

As stated in the previous section, a more detailed PD process is desirable to create a more realistic and changeable representation of the PD process. A problem-solving cycle could be used to generate the activities necessary to develop a product. However, due to the high uncertainty of PD, it is not feasible to generate very detailed activities far in advance [70]. In addition, PD activities are often closely related to a specific subsystem or component [70] (e.g., main body design, structural analysis of the arms). The product architecture and the PD process are strongly linked and in part mirror each other. Therefore, modeling both with a high degree of detail is not necessary, as this would increase the modeling effort. Combining this observation with the logic of a recurring problem-solving cycle consisting of a few distinct activities leads to the following hypothesis.

Hypothesis 1: *If generic process elements are used to recursively generate activities based on the product architecture, then a product development process can be simulated more accurately without significantly increasing the modeling effort.*

5.3 Modeling of Digital Tools

The reviewed M&S methods almost never include digital tools. Especially their impact on how knowledge is created, shared, and used is never considered. This is also true for the baseline model [106]. Because of this, no existing methods can be used or adapted for modeling tools in this thesis, and a new approach is required. Therefore, the following research question will guide the following sections in an effort to fill **Research Gap 2**.

Research Question 2.2: *How can digital tools be abstracted to be represented in M&S methods of product development?*

This section is split into two subsections to handle the observations about the impact of digital tools on PD separately.

5.3.1 Digital Tools as enabling Resources

Observation 2.1 states that *digital tools have powerful capabilities to directly support or enable engineering activities (e.g., design, simulation) that enable faster, higher quality, and more cost-effective development of products, but also require new competencies and adapted processes.* To account for this observation, a simulation model cannot only model tools as resources used by engineers, but must also have an effect on the activity, its result, and the person using the tool.

First, specific tools are required for some activities. Therefore, if a tool does not provide the capability required for an activity, this activity cannot be performed. However, if a tool provides new capabilities, the overall PD process changes due to this new activity being added and potentially impacting downstream activities. For example, if there is no high-fidelity system simulation tool, virtual integration activities (e.g., functional digital mock-ups, digital twins) cannot be performed. If there was a tool with that capability, a detailed virtual prototype could be created to enable the testing of different designs before physical tests begin [64].

Digital tools have become more sophisticated with time, leading to better predictive capabilities [2]. As a result, simulation is faster and more accurate, making its use during PD more effective by delivering better product designs in shorter time frames. However, digital tools still have a performance gap compared to physical tests [130]. Thus, digital tools are used to augment physical testing by reducing the amount of physical testing and possibly also increasing the performance of the product [65, 130].

Lastly, it has been found that engineers must adapt to new tools and that frequent tool changes lead to reduced productivity [2]. Digital tools also require specialized training and digital literacy to be used effectively [17]. Digital literacy is the ability to create, interpret, and communicate information effectively using digital tools, models, and data [66]. Therefore, learning is essential for the use of digital tools, and a ‘competency-capability’ trade-off must be considered when implementing new tools. Learning has already been modeled

through learning curves in many other PD simulation models. They are commonly applied to repeating activities (e.g., [45], [94], [106]) or technical expertise (e.g., [106], [124]).

The following hypothesis is formulated to apply all the above-mentioned ideas.

Hypothesis 2.1: *Digital tools can be modeled as agents that are used by agents of engineers for the execution of activities, where the efficiency and quality of results depend upon the tool's properties and the engineer's digital literacy.*

These tools will be called *Engineering Tools* for this thesis.

5.3.2 Digital Tools as Platforms for Knowledge

In the baseline model, knowledge is only stored by the designer agents themselves and shared among them. However, a significant portion of knowledge is embedded in artifacts, which can be stored, accessed, and exchanged using various tools [59]. **Observation 2.2** states that *digital tools improve the availability, organization, and flow of knowledge and information, ensuring consistency and efficient collaboration through better accessibility, traceability, and exchange between tools and personnel.* To account for this, the use of digital tools to store, exchange, and retrieve knowledge must be considered.

These tools often are platforms acting as a central repository for engineers to access general knowledge, knowledge and insights from previous projects, and, most importantly, information and knowledge about the products being developed throughout their entire lifecycle [16, 59, 64]. Therefore, they could also be modeled as central agents, accessible to all other agents (Engineering Tools, Personnel). In [124], a simple knowledge base was modeled as a central agent that engineers could use to gain competence, therefore, already partially implementing this concept.

Information and knowledge must be transferred between different people due to information dependencies implied in the activity sequence or through product interfaces. Due to the different nature of activities, product elements, or even organizations, different tools

may be used but still require some sort of connection in order to input information from these other tools. The effectiveness of the connections between these tools is highly dependent on their interoperability. High interoperability allows for digital continuity [68]. This leads to fast and error-free data transfer, fostering more efficient and effective processes by enabling concurrent engineering and ensuring that everyone is working with consistent data [19]. Because interoperability is one of the biggest challenges in implementing a digital thread [68], the simulation model of these tools should consider it.

In the baseline model [106], an information agent is used to model communication that occurs probabilistically. Its purpose is purely technical to store the information that is being sent to dependent engineers until that engineer reads the information. Since digital tools and their interoperability influence the efficiency of information transfer, this agent could be changed to also carry properties related to this efficiency and the probability of communication occurring.

By combining the above-mentioned aspects, the following hypothesis is formulated.

Hypothesis 2.2: *Collaborative and knowledge management tools can be modeled as central platforms that connect tools, personnel, and knowledge with each other with varying efficiency depending on interoperability and digital literacy.*

These tools will be called *Engineering Knowledge Management Tools* for this thesis.

5.4 Summary of the Problem Formulation

In this chapter, hypotheses for a holistic M&S framework of PD were formulated based on a selected baseline model. The baseline model was selected by comparing the available approaches with the key characteristics of PD. Figure 5.3 summarizes how the hypotheses were derived on the basis of **Formulation Question 1** and **Research Question 2**. The hypotheses aim to fill the gaps identified in the selected baseline model. The following chapter will present the proposed M&S framework and a methodology to create and test it.

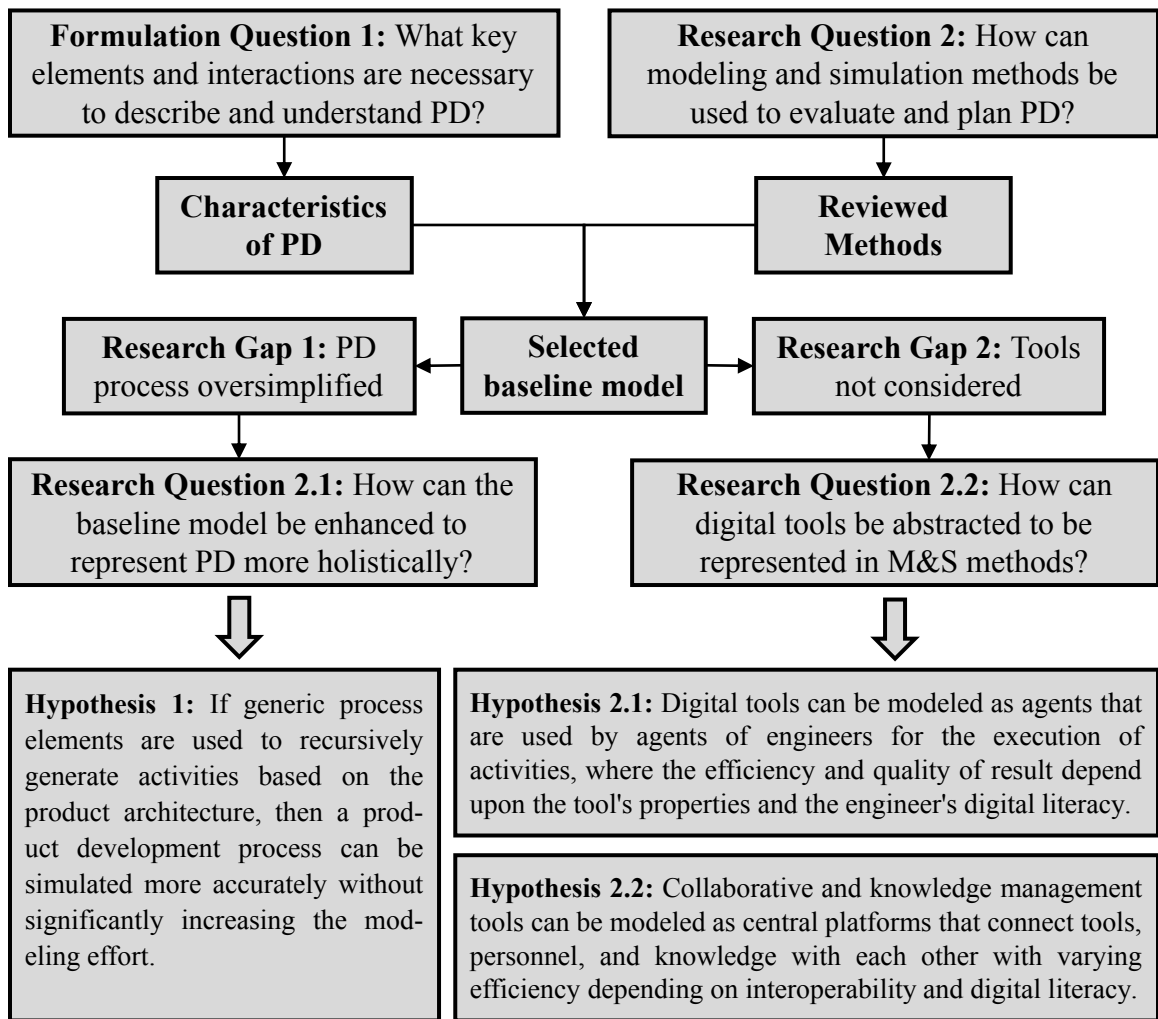


Figure 5.3: Overview of the problem formulation and the resulting hypotheses.

CHAPTER 6

PROPOSED FRAMEWORK AND METHODOLOGY

This chapter presents the proposed M&S framework of complex product development, as well as the methodology proposed to test the hypotheses formulated in the previous chapter. Because this framework extends the work of [106], a description of this baseline model can be found in Appendix A.

6.1 High-level Model Architecture

The proposed M&S framework is based on the structure illustrated in Figure 6.1. This organizes the PD domains and highlights the key interactions that are modeled. The logic of this simulation framework is fully based on the observations made in Chapter 2 and implements the hypotheses proposed in Chapter 5.

First, a product architecture consisting of the hierarchical structure of subsystems and components and the interfaces between components⁷ is combined with a high-level process to generate a detailed process. The procedure for generating this process will be discussed in Section 6.2

The detailed activities are then performed by personnel that are part of the organization. Activities require specific *Engineering Tools* that engineers use for their execution. These activities and tools also require specific competencies. Not fulfilling the required competencies leads to reduced efficiency and quality, but also causes more technical problems, that require collaboration to be resolved. Quality issues, when detected by validation (i.e., simulation, testing) activities, cause rework of activities that can propagate to other dependent activities or product elements.

⁷The original baseline model [106] used functions instead of physical elements to define the product. This was changed to allow for easier selection of the tools required for their development, simulation, and testing.

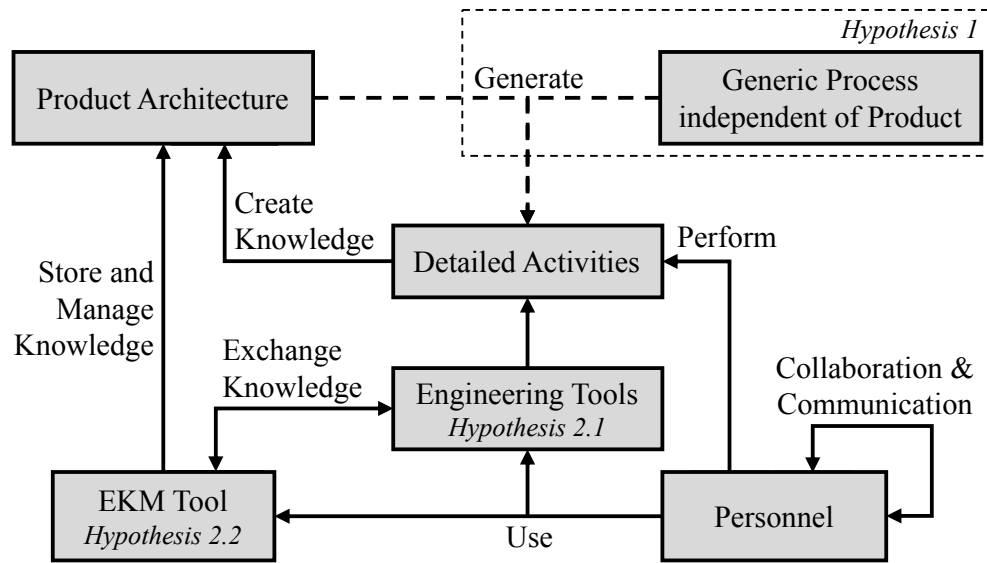


Figure 6.1: Proposed high-level M&S architecture.

Since the product is developed by many engineers, communication has to occur to exchange information about the individual elements of the product. This communication happens between dependent activities and the interfacing product elements. For interfaces, the need for communication is influenced by the complexity of the interface, as well as the knowledge, information, and past experience available to engineers. Collaboration between engineers occurs when problems have to be resolved. The collaboration then increases the engineers' knowledge about other product elements.

Activities generate knowledge and information about the specific element of the product. This is stored in *Engineering Knowledge Management (EKM) Tools*. Engineers use EKM tools to retrieve knowledge or information that is required for their activities. The properties of the EKM tool and the engineers' competency define how efficient the process of sharing, searching, or retrieving knowledge is. EKM tools also facilitate the exchange of information between different tools, with their interoperability influencing the efficiency.

This high-level methodology includes all attributes of PD that have been identified in **Observation 1** and **Observation 2** and therefore presents a holistic view of PD. Because each entity (i.e., product element, engineer, activity, tool) in this framework is represented as agents in the model, their individual states and interactions can be tracked during sim-

ulation. This enables many possibilities for the extraction of data tailored for individual purposes, as defined as a requirement for the M&S framework in Section 3.3.

The following sections will describe aspects related to the hypotheses in more detail.

6.2 Generation of the detailed Process

The proposed high-level PD process can be seen in Figure 6.2. This process is an extension of the problem solving cycle described in Section 5.2.2. Since the framework will focus heavily on how digital tools can be simulated, this PD process is oriented around their use. The basis for this are adapted systems engineering reference processes, such as the W-model proposed by [131] and [132]. These incorporate two “V’s” to account for high-fidelity virtual prototypes and simulation that occurs before physical testing is started. This also relates to model-based enterprise approaches for PD that incorporate two design loops, one virtual and one physical [67].

The activities of this high-level process are generated for every architectural element depending on the type of element. The components are related to the activities *Design*, *Component Simulation*, *Prototype Manufacturing* and *Component Testing*, while the other activities are related to the subsystems. Subsystem activities are generated recursively for every hierarchy level. Furthermore, the capabilities of tools and the responsibilities of per-

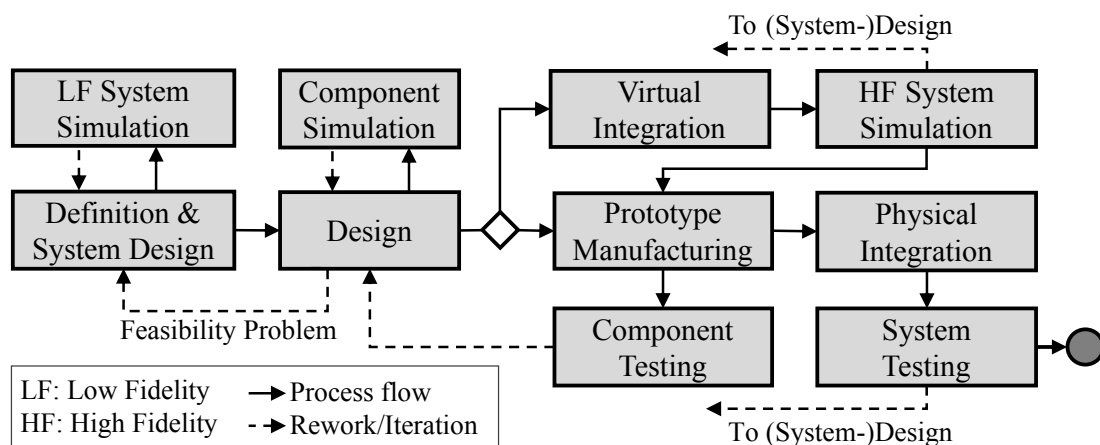


Figure 6.2: High-level product development process used for the proposed framework.

sonnel, which are mapped to activities and architectural elements, influence the generation of activities. If no tool or engineer is capable of performing a specific activity for an element, that activity will not be generated.

In the problem solving cycle, a ‘creation’ task (i.e., system design, design, integration) is followed by a ‘quantification’ task (i.e., simulation, testing). This is also the case in the proposed process. An important step is the evaluation of the quantification results to decide whether the next succeeding activity can be started or if rework must occur. The proposed framework models several types of iteration, depicted as dashed lines in Figure 6.2. Large-scale rework covering multiple activities is possible here, unlike in the baseline model.

It should be stated that the baseline model [106] assigns a single person to one architectural element, an extreme interpretation of the mirroring hypothesis [49]. Due to the many newly added activities, a wider variety of personnel will be required, therefore assigning multiple people to the same architecture element. Therefore, considerations of organizational structures and different responsibilities of personnel are necessary in order to include their effect on the efficiency of communication, knowledge sharing, and decision making [46]. These effects have already been effectively captured by *Task-based Team Interaction* ABS models, for example [102–104], and could be utilized for this framework as well.

To validate **Hypothesis 1**, the simulation results must produce effort distribution curves that more accurately resemble those generated by Norden’s effort model [129], as discussed

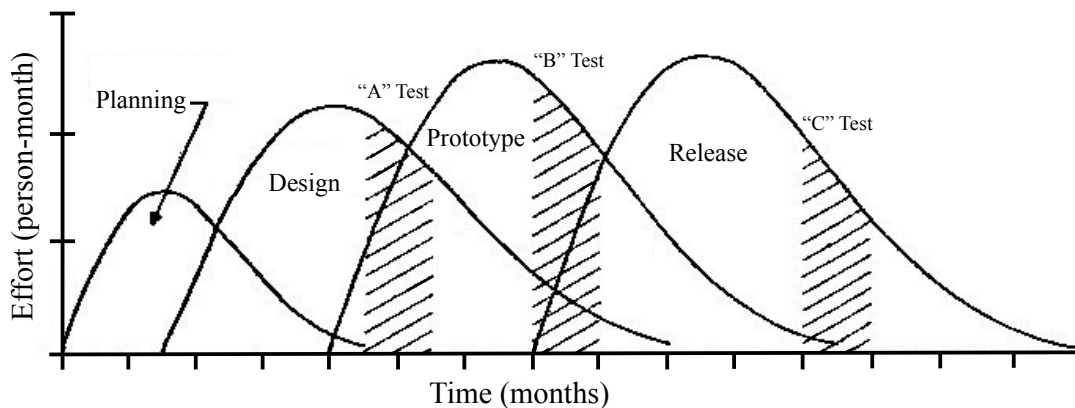


Figure 6.3: Exemplary application of Norden’s effort model to multiple PD stages [129].

in Section 5.2.1. Since multiple stages of PD will be considered, this distribution should emerge for individual stages (Figure 6.3), as originally proposed by *Norden* [129]. An empirical validation will not be possible due to the lack of data. Therefore, this hypothesis will only be validated qualitatively.

6.3 Addition of Digital Tools to the Simulation

In the problem formulation two types of tools were introduced: Engineering Tools and Engineering Knowledge Management (EKM) Tools. In the proposed M&S framework these are represented as agents that can interact with people and other tools, thus influencing the efficiency of work being done and the flow of information. As stated in the problem formulation, the following five dimensions will be considered in the M&S framework.

- *Enablers*: Engineering Tools provide capabilities required for specific activities.
- *Validation*: Varying accuracy of simulation and testing tools.
- *Collaboration*: EKM tools are used to exchange knowledge and information.
- *Interoperability*: Varying efficiency of information exchange between tools.
- *Digital Literacy*: Competencies for tools are required for their efficient use.

To implement this, some interactions and properties have to be defined. Tools must be associated with specific activities and product elements to define their capabilities to enable the detailed activities mentioned in the previous section. In addition, engineers must be able to interact with EKM tools to provide and retrieve knowledge and information.

To define how good validation activities (i.e., simulation, testing) are, an accuracy metric is used. This measures the predictive capability of specific tools. A high predictive capability leads to better forecasts of a systems behavior [133], therefore leading to better detection of errors and the initiation of necessary rework. Thus, a high-accuracy simulation tool would reduce the need for physical testing because issues are detected before physical testing starts, which has been observed as a major advantage of digital tools in PD [14,

65]. To simulate this effect, the tools used for physical testing (i.e., test equipment, test facilities) will also be included in the simulation framework.

The interoperability between tools defines how good knowledge or information from one tool can be used by a different tool both syntactically and semantically [69]. This metric would therefore influence how fast and often information can be exchanged and used by other tools. Here, interoperability represents a highly abstracted concept of information exchange efficiency between tools through the use of standards and the use of automation. Low interoperability would mean that a lot of manual effort is required to prepare and transform data. This also affects the completeness and availability of information and knowledge in EKM tools.

Lastly, the digital literacy required to use the tools will be considered. The baseline model [106] already compares the knowledge requirements of the product elements with the expertise of the engineers to determine their efficiency, problem rates, and the quality of the product design. In addition, learning to increase expertise and knowledge is modeled through learning curves. This approach can therefore also be applied to digital tools by defining the ease of use of the tools and the tool competency (i.e. digital literacy) of engineers. A mismatch would then lead to reduced efficiency and more problems occurring.

Because the proposed framework is still conceptual, the proposed properties of the tools are very abstract. In this thesis, no methods for quantifying any of these properties are provided, as the goal is to qualitatively demonstrate how the digital tools used during PD could be abstracted for the purpose of simulating the PD process.

To do this, different scenarios will be analyzed. If these scenarios exhibit the behavior expected from digital tools, the hypotheses can be verified. To validate **Hypothesis 2.1**, highly capable Engineering tools must demonstrate a reduction in physical testing and overall PD lead time. To validate **Hypothesis 2.2** less iteration due to interface incompatibilities has to occur when highly interoperable tools are used because the information used is more consistent.

In addition, both Engineering and EKM tools have to enable the simulation of the ‘competency-capability’ trade-offs. This should be understood as the trade-off of implementing new more capable tools that require new or better competencies, which leads to lower productivity when no training is provided.

6.4 Notional Case Study for Testing the Framework

The framework will be applied to a case study in order to perform various tests to validate the hypotheses. It has to be relatively simple to implement to ensure that it remains within the scope of this thesis without requiring excessive time. However, sufficient complexity is also required to validate the hypotheses, especially with respect to the emergent behavior of PD.

The baseline configuration for the notional case study is structured as follows. A simple drone has been selected as the product (Figure 6.5), as this is a multidisciplinary product with multiple different dependencies and interfaces, but also has a manageable number of subsystems. The organization (Figure 6.4) is divided into four teams that make up the entire project team. Various tools with different capabilities have also been defined (Table 6.1). The current selection of tools presents a rather ‘traditional’ engineering process without any highly capable simulation tools. Because the current framework does not explicitly include procurement, a ‘Supplier’ is defined as a tool to act as a proxy to generate activities related to prototype procurement. More detailed information on the defined configuration can be found in Appendix C.

A notional configuration was chosen due to the limited data available. Due to the complex nature of this problem, a lot of hard-to-access data would be required. For this, process mining [84] would most likely have to be used, which is currently beyond the scope of this thesis. Companies willing to share this sensitive data would also have to be found. Therefore, this thesis will not be able to validate the framework against empirical data but can only use general trends described in the PD literature to validate the intended behavior.

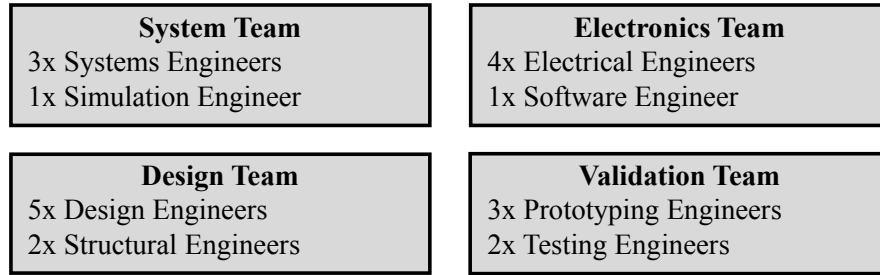


Figure 6.4: Teams and engineers defined for the baseline organization.

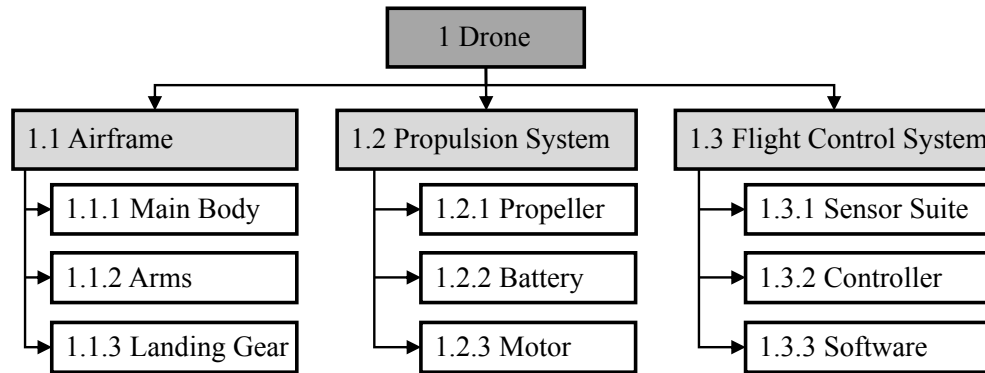


Figure 6.5: Hierarchical product architecture of the notional drone.

Table 6.1: Tools defined for the baseline organization and their capabilities.

| Tool | Capabilities |
|-------------------------|--|
| MBSE | System Design (1, 1.1, 1.2, 1.3) |
| Basic System Simulation | LF System Simulation (1, 1.2, 1.3) |
| MCAD | Design & Virtual Integration (1.1, 1.1.1, 1.1.2, 1.1.3, 1.2.1) |
| ECAD | Design (1.3.2) |
| IDE | All Activities (1.3.3) |
| FEA | All Simulation Activities (1.1, 1.1.1, 1.1.2, 1.1.3, 1.2.1) |
| Circuit Simulation | Component Simulation (1.3.2) |
| Manufacturing Equipment | Prototyping (1.1.1, 1.1.2, 1.1.3, 1.2.1, 1.3.2) |
| Assembly Equipment | Prototyping (1, 1.1, 1.2, 1.3) |
| Universal Test Machine | Testing (1.1, 1.1.1, 1.1.2, 1.1.3, 1.2.1) |
| HIL Equipment | Testing (1.3, 1.3.2) |
| System Testing Facility | Testing (1, 1.2) |
| <i>Supplier Proxy</i> | Design & Prototyping (1.2.2, 1.2.3, 1.3.1) |

To validate the framework, the goal of the implementation will be to replicate the behavior and attributes captured in Chapter 2 and also provide enough insights to fulfill the modeling objective and purpose stated in Chapter 3. More specifically, this is related to **Observation 1.2** and **1.3** to validate **Hypothesis 1** (i.e., the process logic), as well as **Observation 2.1** and **2.2** to validate **Hypothesis 2.1** and **2.2** respectively (i.e., digital tools). To demonstrate the capabilities of the proposed framework, different scenarios will be defined that relate to an organization's potential efforts to increase PD efficiency. These will be described in more detail in Section 8.2.

The next chapter will provide a detailed description of how the proposed framework was implemented and set up to be validated.

CHAPTER 7

IMPLEMENTATION OF THE M&S FRAMEWORK

This chapter will present the developed simulation model and implementation. The model was built entirely with Python. This enables the implementation of custom functions, logic, and data extraction that might be more complicated in commercial ABM software. In addition, the creation of a customized dashboard is also possible to visualize detailed data. Although building the initial framework is time-consuming, it allows for high flexibility to implement exactly what has been proposed in the previous chapters. The entire code can be found on GitHub (see Appendix D).

The implementation was already partially published in [134] as a preliminary simulation framework and is based on the selected baseline simulation model [106], described in Appendix A. This chapter extends the partial implementation to fully realize the proposed framework. The following sections are organized and guided by the proposed high-level modeling logic presented in the previous chapter. First, the product architecture will be discussed, as it guides the flow of the simulation together with the process defined in Section 6.2. After that, the simulation logic centered around the agents of engineers will be discussed. All other entities of the simulation will also be discussed in more detail in that section. Supplementary information on the defined parameters and detailed input data used for the notional case study is provided in Appendix B and Appendix C, respectively.

7.1 Product Architecture

The product architecture defines the structure of the product that is being developed. The hierarchy guides the decomposition into subsystems and components, and interfaces define how the components interact and connect to each other. Interfaces are defined through different types and a corresponding severity. The complete list of the types used can be found

in Table B.1 in Appendix B. The interfaces for the product architecture of the notional case study are defined in a design structure matrix (DSM) that can be found in Figure C.2 in Appendix C. A symmetric DSM was chosen for this thesis, but is not required.

A metric based on important product complexity drivers (i.e., hierarchy, interfaces, multidisciplinary) [3], proposed by [37], is used to characterize the complexity of individual product elements and their interfaces.

The complexity of any product element is defined by the amount of knowledge gained from formal education and training that is needed for its development [106]. This knowledge requirement is represented as a vector \vec{K}_R , where each value k_n defines the requirement for a specific knowledge domain (e.g., mechanical, electrical, etc.). Table B.2 in Appendix B lists all defined knowledge domains. The severity of this knowledge requirement is measured on a scale from 0 to 3. Equation 7.1 is used to calculate the technical complexity TC of a product element based on its knowledge requirements, where N is the number of knowledge requirements [37].

$$TC = \sqrt{\frac{1}{N} \sum_{n=1}^N k_n^2} \quad (7.1)$$

Integrating elements is more complex with many severe interfaces and when the mutual understanding of engineers decreases due to the need for different knowledge [106], thus increasing the need for coordination. Therefore, the integration complexity IC is calculated from the severity of the interface IS and the knowledge difference KD of the elements, as expressed in Equation 7.2.

$$IC = 0.5 \cdot IS \cdot KD \quad (7.2)$$

The constant 0.5 is added to distribute the complexity between the two elements of the interface. The interface severity IS is defined as the sum of all severity values for the different types of interfaces, divided by the highest possible interface severity IS_{max} . The knowledge difference KD between elements is defined as the angle between their knowl-

edge vectors \vec{K}_R in the multidimensional knowledge space, which is scaled with the upper limit of the knowledge scale r to maintain the same order of magnitude as the technical complexity [37], as stated in Equation 7.3.

$$KD = r^{\sin \Theta} = r \sqrt{1 - \left(\frac{\vec{K}_{R,1} \cdot \vec{K}_{R,2}}{\|\vec{K}_{R,1}\| \|\vec{K}_{R,2}\|} \right)^2} \quad (7.3)$$

The complexity of the elements and their interfaces is used to define the effort required for activities (Section 7.2.1), the amount of information flow required between interfacing elements (Section 7.2.2), and the difficulty to resolve technical problems (Section 7.2.3).

7.2 Detailed Simulation Logic

The high-level simulation logic has already been discussed in Section 6.1. This logic and the generated process (see Section 6.2) guide the flow of the simulation on a high level by defining the sequence of activities and the information flow that must occur. The logic is extended by the detailed behavior of the individual agents of engineers⁸, depicted as a state machine diagram in Figure 7.1.

Agents perform technical work when tasks are ready. The competence of the agents and the tools used influences the outcome of these tasks influencing the quality of the product and the accuracy of validation results. Technical work will be discussed in more detail in Section 7.2.1. During or after technical work, multiple events are possible and occur probabilistically. The probability of events occurring is influenced by the properties of the engineers and tools.

Information about dependent elements or preceding activities could be needed. If that information is not available on the *Engineering Knowledge Management* (EKM) Tool, it is requested. Once information is available, it is processed and the compatibility of interfaces or the feasibility of requirements is checked. The processing of information during the

⁸For simplicity the agent of an engineer will be called ‘agent’ from here on.

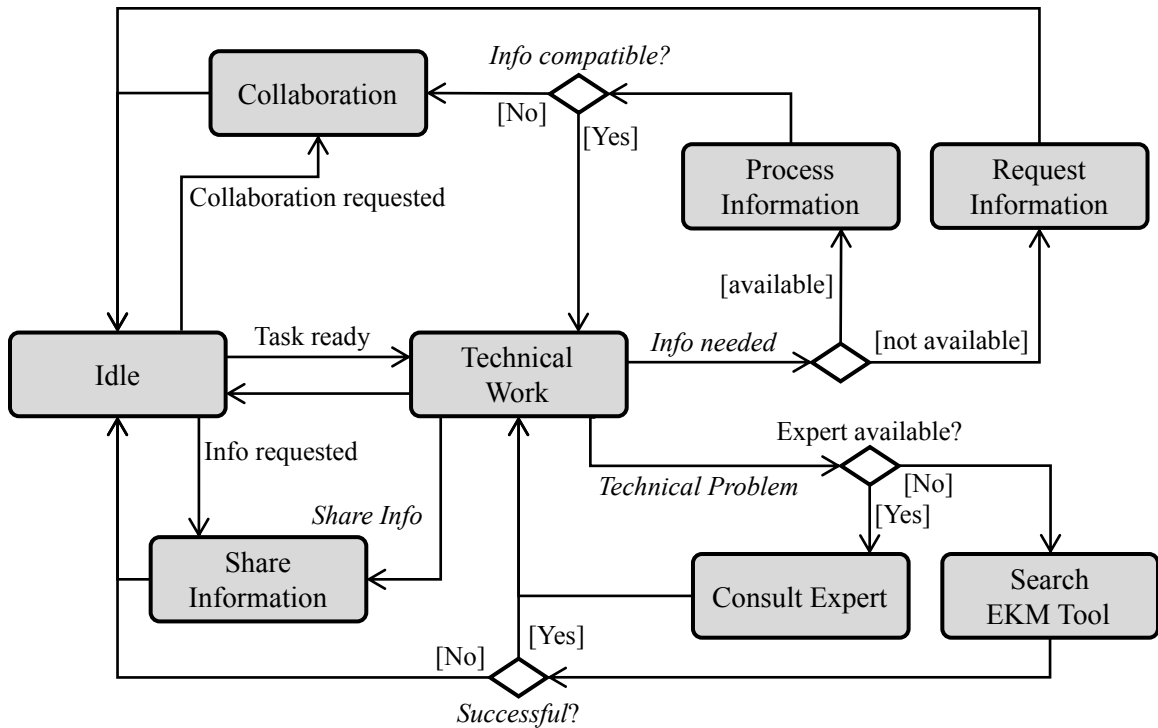


Figure 7.1: State machine diagram for the behavior of engineer agents in the simulation model (*Italics: probabilistic events*).

simulation is discussed further in Section 7.2.2.

If the checked information does not cause any problems, technical work can be continued. However, if problems occur, collaboration is initiated to resolve the problem. Similarly, if technical problems due to missing expertise occur during technical work, an expert is consulted to resolve the problem. If no expert is available, the EKM tool is searched for knowledge. If this is successful, technical work is continued. If not, consultation with an expert is requested. The resolution of problems through collaboration, consultation, and the EKM tool will be discussed in more detail in Section 7.2.3.

7.2.1 Technical Work

Technical work refers to the effort applied to the activities generated from the product architecture and the high-level process (see Section 6.2).

Effort of Activities

To account for uncertainties, the effort required for individual activities is estimated using triangle distributions, as is common practice in the PD literature [73]. To account for the increased effort required for complex products, the distribution $TRI(a, b, c)$ is multiplied by the technical complexity TC of its corresponding product element, as stated in Equation 7.4. The nominal effort E_{act} required for an activity is sampled only once at the beginning of the simulation.

$$E_{act} = TRI(a, b, c) \cdot TC \quad (7.4)$$

Each activity is modeled as a series of sequential tasks, as proposed in the baseline model [106]. This is done to discretize the occurrence of events during the simulation. Tasks are assumed to require a constant amount of effort. Therefore, the number of tasks n_t is calculated based on the task duration E_{td} , as stated in Equation 7.5. To avoid rounding errors, Equation 7.6 calculates the nominal task effort E_{tn} based on the number of tasks. The learning effects resulting from the repetition of a task are modeled in Equation 7.7. The effort is reduced using learning curves modeled with the power model [135], where l is the learning rate and n_{rep} is the number of repetitions of the task. The learning rate represents how much the nominal effort of a task is reduced after each repetition. Consequently, a lower learning rate corresponds to a larger reduction in effort.

$$n_t = \text{nint}(E_{act}/E_{td}) \quad (7.5)$$

$$E_{tn} = E_{act}/n_t \quad (7.6)$$

$$E_t = E_{tn} \cdot n_{rep}^{\log_2 l} \quad (7.7)$$

The lower limit a , upper limit b and mode c of the triangle distributions, as well as the learning rates l used for individual activities can be found in Table C.2 in Appendix C.

Working on Tasks

When agents are idle, they check what tasks can be performed by them. For a new task to start, all its predecessors have to be completed. This is defined through the activity network created from the high-level process and the product architecture.

When a task within the responsibilities of an agent is ready, the agent starts working on that task. If multiple tasks are ready for one agent, a selection is made based on the priority of the tasks. The priority is derived from the importance of the product element associated with the task and the urgency of the task, which is the amount of time that the task has been ready, as proposed by [103]. Tasks that have already been started are finished before a different task is started. Tasks can only be interrupted for collaboration with other agents.

Agents can only have one responsibility for a *System Design* or *Design* activity. This was done for implementation reasons to make modeling the information flow between agents on interfacing elements easier. For other activities, an agent can have multiple responsibilities. All defined responsibilities can be found in Table C.3 in Appendix C.

Equation 7.8 calculates the efficiency of an agent while working on a task, where A_C is the agent's competence, A_{TC} is the agent's tool competence, and T_p is the productivity of the tool. Low efficiency results in an increased effort required to complete a task.

$$A_{eff} = A_C \cdot A_{TC,Eng} \cdot T_p \quad (7.8)$$

The tool productivity defines how much effort while working on tasks is made more efficient through, for example, automated workflows or assistance functions.

The agent's competence is defined by the mismatch of the agent's technical expertise with the knowledge required for the development of a product element. Therefore, technical expertise is defined by the same knowledge vector that is used to define the knowledge requirements of product elements. It refers to an engineer's knowledge in various domains and determines the ability to perform a certain task. Equation 7.9 calculates the agent's competence [106], where k_n is the knowledge requirement for knowledge domain n , a_n is

the agent's expertise in knowledge domain n , M is the set of knowledge domains that satisfy $k_n > 0$, and m is the size of the set M . Through learning processes, competence can be increased throughout the simulation (see Section 7.2.3).

$$A_C = \frac{1}{m} \sum_{n \in M} \min \left\{ 1, \frac{a_n}{k_n} \right\} \quad (7.9)$$

Tool competence is defined as the ratio of digital literacy to tool usability, as stated in Equation 7.10, where $A_{DL,Eng}$ is the agent's digital literacy for the Engineering tool used and T_{Usab} is the tool's usability. Digital literacy is a generic term used here to define an agent's proficiency in using digital tools, models, and data. The usability refers to the ease with which a tool is used and defines the amount of training required to use a tool. A large value implies that more training is required.

$$A_{TC,Eng} = \min \left\{ 1, \frac{A_{DL,Eng}}{T_{Usab}} \right\} \quad (7.10)$$

The tool competence and technical competence are used to define the efficiency, as discussed here, as well as the efficacy of the agents, which will be discussed in the following sections. Excessive expertise or digital literacy can partially increase efficiency beyond one, resulting in faster than nominal task completion. The efficacy cannot exceed 1.

All properties of the agents and tools can, respectively, be found in Table C.4 and Table C.6 in Appendix C.

Once a task has been completed by fully applying the total required effort E_t , events can be triggered, and the succeeding task can start. What events are triggered depends on the type of activity. All possible events will be discussed in the following sections.

Quality of Work

System Design and *Design* activities define the quality of the product elements. The definitions used to model the quality are all based on the baseline model [106] and adapted to

include the impact of system design activities. The quality of product elements and interfaces is modeled as the probability of successful validation, discussed later in the section. Figure 7.2 illustrates the different quality types defined for product elements and how they relate to each other.

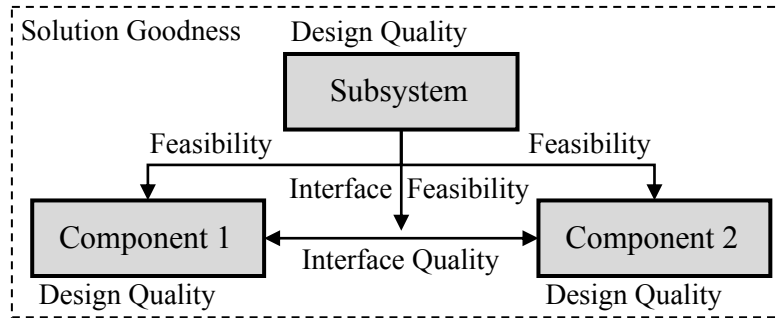


Figure 7.2: Quality types defined for product elements.

The competence A_C (Equation 7.9) of an engineer defines how well the knowledge requirements of a product element are matched. Fewer technical mistakes are expected when the engineer's competence meets more requirements. Therefore, the design quality Q_D of product elements is equal to the competence of the agent designing it [106].

During *System Design* activities, requirements are established for subsystems and components. The feasibility of these requirements defines the upper limit of the design quality. If an engineer has a lot of experience with a product, more feasible and balanced requirements are expected to be defined. Therefore, the feasibility Q_F is defined to be equals to the product knowledge PK about the specific product element. Product knowledge refers to the experience an engineer has with a product through similar previous projects. PK is represented as values between 0 and 1. The initial product knowledge of each agent is determined by multiplying the novelty of the product element with the engineer's prior experience. Throughout the simulation, product knowledge is increased through collaboration, as further described in Section 7.2.3.

The work on interfaces requires an understanding, i.e., product knowledge PK , of the dependent elements, as well as the exchange of information to align individual designs with

each other. More experience results in better anticipation of how an interfacing element will be designed, therefore increasing the interface quality Q_I [106]. Because interfaces are already partially defined through requirements set during *System Design* activities [6], the required exchange of information is reduced with increasing feasibility of the interface. The interface feasibility Q_{IF} is equal to the system designer's average product knowledge PK about the interfacing elements. The interface quality is then defined as

$$Q_I = PK \cdot (Q_{IF} + (1 - Q_{IF}) \cdot I_{rel}) \quad (7.11)$$

where I_{rel} is the relevant information used to design the interface. Section 7.2.2 will further discuss the definition of information in the context of the simulation.

The integrated goodness of a system or the complete product Q_G defines how well individual elements of that system are designed with respect to the elements they interface with. The goodness of a system is limited by the weighted average of the level of goodness of its subsystems. The goodness of the subsystems is defined in Equation 7.12, where i are all subsystems or components of the integrated system, w_i is the importance of subsystem i , and $Q_{G,i}$ are the individual levels of goodness. The importance is defined as the degree to which a subsystem contributes to satisfying the overall product requirements. The importance could be quantified through a QFD analysis or other comparable methods.

$$\overline{Q_{G,sub}} = \frac{\sum_i Q_{G,i} \cdot w_i}{\sum_i w_i} \quad (7.12)$$

The goodness of a product element is then modeled using Equation 7.13 [106]. Q_D is the design quality of the element, $\overline{Q_I}$ is the average interface quality of all interfaces in the subsystem, and α_g and β_g are shape parameters.

$$Q_G = \frac{Q_D \cdot \overline{Q_{G,sub}}}{1 + \alpha_g \cdot e^{-\beta_g \cdot \overline{Q_I}}} \quad (7.13)$$

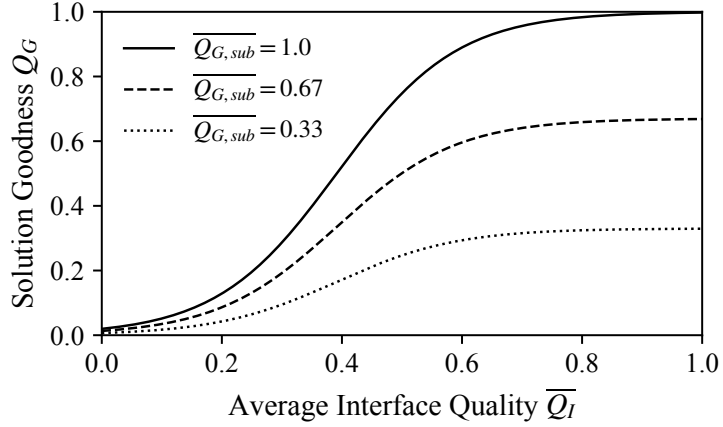


Figure 7.3: Function shape of Equation 7.13 ($Q_D = 1$, $\alpha_g = 50$, $\beta_g = 10$) to model the impact of the interface quality on the solution goodness of an integrated product element.

Figure 7.3 depicts the shape of this function by varying the average interface quality. An S-shaped function was chosen based on observations that slower quality changes are expected at the beginning and end of PD [136].

The quality of the interfaces define how well the components of the subsystem integrate with each other, especially when the difficulty to integrate systems is high. Therefore, the weighted average of the interface qualities $\overline{Q_I}$ of a subsystem is defined in Equation 7.14, where i are the interfaces within the subsystem, $Q_{I,i}$ is the quality of interface i , and IC_i is the integration complexity of interface i .

$$\overline{Q_I} = \frac{\sum_i Q_{I,i} \cdot IC_i}{\sum_i IC_i} \quad (7.14)$$

Validation through Simulation and Testing Activities

The purpose of validation activities is to check the quality of a design. *LF System Simulation*, *Component Simulation*, and *Component Testing* activities check the design quality Q_D . *HF System Simulation* and *System Testing* activities check the solution goodness Q_G of the product or a subsystem that includes all its subsystems, components, and interfaces. Interfaces with other components and subsystems are partially validated, depending on the integration complexity IC and the interface feasibility Q_{IF} , which define the difficulty of

testing an external interface and the completeness of the requirements set for an interface.

Since the tools used to perform the validation activities can vary in accuracy, the measured quality is changed based on this accuracy. The measured quality \hat{Q} is defined as

$$\hat{Q} = 1 - (1 - Q)^{1/Acc} \quad (7.15)$$

where Q is the actual quality and Acc is the accuracy of the agent-tool pair working on the validation task. For $Acc < 1$, the measured quality is greater than the actual quality. Figure 7.4 visualizes this relationship in more detail. The accuracy is defined as

$$Acc = T_{Acc} \cdot A_{TC,Eng} \cdot A_C \quad (7.16)$$

$$Acc' = 0.5 \cdot (T_{Acc} \cdot A_{TC,Eng} + A_C) \quad (7.17)$$

where T_{Acc} is the accuracy of the tool itself, $A_{TC,Eng}$ is the agent's tool competence (Equation 7.10) and A_C is the agent's technical competence (Equation 7.9). The accuracy of the validation tool describes its predictive capability [133], where 1 would correspond to perfect predictions, and values close to 0 would correspond to very poor or limited predictions. Equation 7.17 is used when a tool is not required, but can enhance validation (see Section 7.2.3).

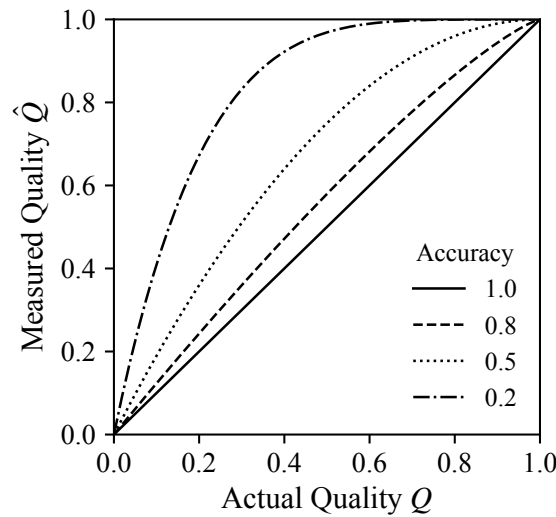


Figure 7.4: Function shape of Equation 7.15 to model the impact of the tool accuracy on the measured quality.

The probability of ‘successful’ validation is then defined in Equation 7.18, where \hat{Q} is the measured quality and T_Q is the quality target.

$$P_V = \min \left\{ 1, \frac{\hat{Q}}{T_Q} \right\} \quad (7.18)$$

Thus, when the accuracy is low, validation is more likely to ‘succeed’ and quality issues are not detected. Later (physical) validation activities can still detect these quality issues, leading to more rework. Quality is checked after every completed validation task.

If validation fails, rework of the affected activities must occur. For *LF System Simulation*, *Component Simulation* and *Component Testing*, the direct *System Design* or *Design* activity of the product element is reworked. For *HF System Simulation* and *System Testing*, the system itself or any of its subsystems, components, or interfaces can be the cause of failed validation. Therefore, the activity that must be reworked depends on the product element that failed validation. The portion of work that must be reworked is equal to $1 - Q$. Furthermore, the quality gains that can be made through rework are limited by $(1 - \hat{Q}) \cdot A_C$.

All downstream activities working on hierarchically dependent product elements are reset as a result of this iteration (first-order rework). Change propagation to interfacing elements (second-order rework) is currently not considered.

7.2.2 Processing Information

Information (i.e., drawings, data, reports, models) is generated by agents working on activities. Information is assumed to be created linearly with the progression of an activity. The exchange of this information is essential in PD, because people work on different activities that have to be aligned with each other. All information, once shared, is stored in the EKM tool, where it can be accessed by other agents and tools when needed. If information is not available when it is needed, it is requested from the agent responsible for creating it. The need for information about dependent elements is based on the integration complexity, the goodness of the interface requirements, and the amount of missing information.

Information Completeness and Correctness

Every agent who uses information about product elements has properties associated with that information to define the completeness of the information (I_{comp}) and the correctness of the information (I_{corr}). The completeness measures how much of required information is available to that agent, and the correctness measures how up-to-date that information is. The amount of correct information that is relevant for technical work is defined as

$$I_{rel} = I_{comp} \cdot I_{corr} \quad (7.19)$$

Upon receiving information about a product element, the completeness and correctness are updated to match the information source. The source of information is either the agent that created this information or the EKM tool. Using more relevant information during the design of components increases the interface quality as defined in Equation 7.11.

Upon rework, some information about the product elements becomes obsolete. Therefore, the amount of correct information used by the agents is reduced. For the agent responsible for designing the element, the information completeness is reduced by the amount of rework. For all other agents and the EKM tool, the information correctness is reduced by the percentage of rework. If a subsystem is reworked, the same reduction of information occurs for all of the subsystem's hierarchically dependent elements. The reduced correctness leads to quality problems on interfaces if updated information is not exchanged. Once an agent receives new information, the inconsistencies are resolved.

Information Exchange

Information exchange is required due to information dependencies between interdependent product elements that are developed concurrently and between sequentially dependent activities. In addition, information is also needed for the verification of designs based on requirements set during system design.

For a new activity to start, relevant information from predecessors (*System Design, Design*) or information about a design (all other activities) must be received. The nominal effort required to process information before starting the activity is modeled with

$$E_{I,n} = e_{info} \cdot E_{act} \cdot \Delta I_{rel} \quad (7.20)$$

where e_{info} is the information handling effort per unit of information, E_{act} is the nominal effort of the activity the information stems from, and ΔI_{rel} is the difference between relevant information that has already been processed and the information needed. Because design verification is not modeled as a separate activity in the simulation, e_{verif} is introduced to increase the effort of verification work, as it is assumed that this requires additional effort.

The actual effort to process information is defined in Equation 7.21, where T_I is the Engineering tool's interoperability with the EKM tool, $A_{TC,EMK}$ is the agent's competence for the EKM tool, and T_p is the EKM tool's productivity. The EKM tool competence is calculated based on Equation 7.10 by substituting for the properties of the EKM tool. Interoperability in this context measures how effectively data from one tool can be used by another. This refers to the use of standards, APIs, or automated data exchange capabilities within a digital thread [68]. A value of 0 indicates that there is no data compatibility, while a value of 1 represents seamless data sharing that requires no effort. Most tools fall between these extremes, meaning that some manual effort is required to exchange and utilize data.

$$E_{I,a} = \frac{E_{I,n} \cdot (1 - T_I)}{A_{TC,EMK} \cdot T_p} \quad (7.21)$$

Interfacing elements are developed concurrently. Therefore, not all information about dependent elements is available at the start of a design activity. This information is exchanged throughout the duration of the design activities. Figure 7.5 depicts the interactions between agents and tools while working on interdependent elements. Only relevant information is exchanged. Complex dependencies require more information exchange and

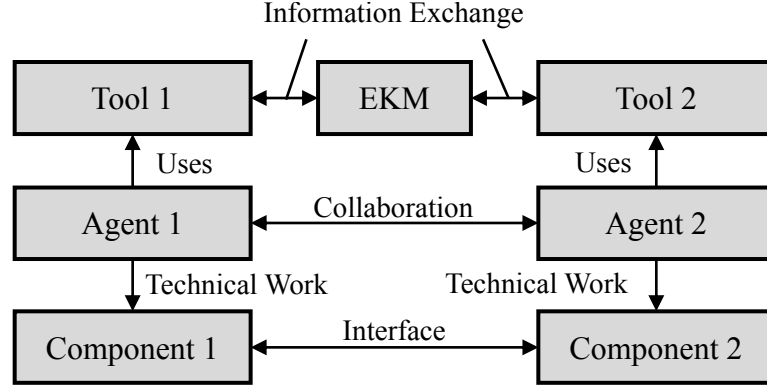


Figure 7.5: Information exchange between agents designing interdependent elements.

coordination [137]. Therefore, the fraction of information that is exchanged between interdependent product elements is modeled as the ratio of the interface's integration complexity IC to the element's total complexity, defined as $(TC + IC)$.

Information exchange between interdependent elements and the verification of designs, while working on *System Design* and *Design* activities, is more likely if the agent is proficient in using the EKM tool ($A_{TC,EKM}$). This likelihood increases further if the interoperability (T_I) of the specific Engineering tool with the EKM tool is high and if the EKM productivity (T_p) is high, as defined in Equation 7.22. The assumption for this is that if information can be exchanged easier and faster, it will be done more often. $P_{I,b}$ and $P_{I,f}$ are tuning parameters to adjust the frequency of information exchange.

$$P_I = P_{I,b} + P_{I,f} \cdot A_{TC,EKM} \cdot T_I \cdot T_p \quad (7.22)$$

If information is needed or has been requested, the sharing and processing of information is not modeled probabilistically. Here, only the priority of the respective tasks and requests is considered to determine when information is exchanged.

7.2.3 Collaboration and Consultation

Collaboration in PD is important to align engineers working on different parts of the product with each other [54], but also to transfer knowledge from experts to novices [61].

Three types of collaboration are modeled: Consultation of an expert due to technical problems and collaboration between agents to resolve interface or feasibility issues. Once any of these problems occur, collaboration is requested from an expert or responsible agent. If the agent is available, the collaboration can begin immediately. When this is not the case, collaboration is requested and will be started once the request is accepted. The priority of requests is defined the same as that for tasks. Thus, when an agent selects what to work on, the highest priority request or task is chosen.

The purpose of consultation and collaboration is to resolve problems and, consequently, to increase the knowledge of an agent. This models the learning process of engineers through the exchange of tacit knowledge. Increased knowledge then leads to fewer problems and a higher quality of the product. The following three subsections will introduce the individual problems that cause collaboration, and the last subsection introduces the equations used to model knowledge improvement.

Technical Problems

A technical problem occurs due to lack of expertise in a knowledge domain required for a technical task. The problem forces the agent to stop working and seek help. An exponential distribution defines the time interval between technical problems. Equation 7.23 calculates the rate at which technical problems occur [106], where k_n is the knowledge requirement for knowledge domain n , a_n is the agent's knowledge of knowledge domain n , and M is the set of knowledge domains that satisfy $k_n > 0$. If all knowledge requirements are met, technical problems will not be encountered.

$$\lambda_{tp} = \max_{n \in M} \left(1 - \frac{a_n}{k_n} \right) \quad (7.23)$$

To help resolve the technical problem, the agent consults an available expert. In the simulation, an expert is defined as an agent that has a knowledge level higher than the required knowledge level for the knowledge domain that causes the problem. The consul-

tation improves the agent's expertise, increasing their efficiency and efficacy.

If no expert is available, the agent attempts to use the EKM tool to retrieve general knowledge. This is modeled with reduced efficiency (η_{EKM}) and a probability of failure depending on the digital literacy of the agent $A_{DL,EKM}$ and the completeness of the EKM tool. If the search on the EKM tool failed, a consultation meeting with an expert is requested. The request is then accepted depending on the expert's other priorities.

Feasibility Problems

A feasibility problem is encountered during the design or validation of a product element. Regardless of how good a product element is designed, it may fail to meet the required quality if the underlying requirements are not well-defined. The probability of this occurring after the completion of a task is $1 - \hat{Q}_F$. \hat{Q}_F is the measured feasibility of a product element, calculated using Equation 7.15 and 7.17. The tool accuracy of design tools is related to simulation capabilities within them.

A feasibility problem requires the *Design* activity to be stopped until the feasibility is improved. For this, the agent responsible for specifying the requirements and the agent responsible for the design collaborate. This increases the system designer's product knowledge of the lower-level element. After the collaboration is finished the *System Design* activity is reworked, improving the feasibility of the requirements and allowing the paused activity to be continued.

Interface Incompatibilities

After receiving information on interdependent elements, that information is checked to verify its compatibility with the interface. The probability of an element failing this check is $1 - Q_I$, where Q_I is the quality of the interface. Validation activities can also detect interface incompatibilities.

The verification of interface information is done by the system designer of the subsys-

tem that contains the interface and the designer of the dependent element. The agent designing the element only checks the incoming information for its compatibility. The system designers, however, verify the alignment of both elements with the defined requirements.

The probability of the system designer to detect incompatibilities is higher when high-quality requirements are defined for the interface (I_{IF}). Additionally, the probability is reduced based on the accuracy of the agent-tool pairing performing the verification. Here, accuracy relates to the fidelity, completeness, and traceability of models created to perform the verification. The accuracy and resulting reduced probability of detecting issues is calculated using Equation 7.15 and 7.17.

When an incompatibility is detected, the agents responsible for designing the interface collaborate to learn about each other's designs and resolve the issue. This increases their product knowledge of the dependent element. The effectiveness of collaboration triggered by the verification by a system designer is based on the quality of the requirements set for the interface (I_{IF}).

After the collaboration, the product element that caused the interface incompatibility is reworked. The percentage of rework is based on the amount of information that was verified and the interface quality: $I_{new} \cdot (1 - Q_I)$. The latest information on the product element is also exchanged before rework starts.

Knowledge Improvement

The effect of consultation and collaboration is the improvement of general knowledge (expertise) or product knowledge. The improvement is modeled as an exponential rise that starts at the initial knowledge (a_n^{init} or PK^{init}) and the knowledge increases with consultation or collaboration effort (E_{cnslt} or E_{collab}). During consultation, knowledge increases towards the expert level X_n . For collaboration, the maximum value is 1, since product knowledge is increased. Equation 7.24 calculates the improvement of general knowledge, while Equation 7.25 calculates the improvement of product knowledge [106]. The effi-

ciency of consultation or collaboration (η_{cnslt} or η_{collab}) is the complexity resolved per hour. The learning efficiency is reduced according to the complexity of the element (TC) or interface (IC) due to the increased difficulty of solving problems in a complex context.

$$a_n^{New} = \frac{X_n}{1 + \left(\frac{X_n}{a_n^{init}} - 1\right) \cdot e^{-\frac{\eta_{cnslt} \cdot E_{cnslt}}{TC}}} \quad (7.24)$$

$$PK^{New} = \frac{1}{1 + \left(\frac{1}{PK^{init}} - 1\right) \cdot e^{-\frac{\eta_{collab} \cdot E_{collab}}{IC}}} \quad (7.25)$$

Figure 7.6 illustrates the improvement of product knowledge with collaboration effort for different integration complexities. The shape of the function for consultation on general knowledge is the same, the only difference being the change in scale depending on the level of expert knowledge X_n and the initial level of knowledge a_n^{init} .

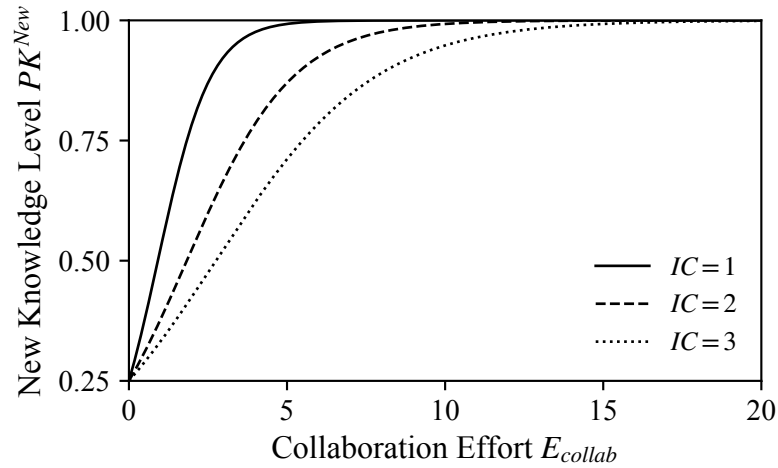


Figure 7.6: Function shape of Equation 7.25 ($PK^{init} = 0.25$, $\eta_{collab} = 0.5$) to model the improvement of knowledge through collaboration.

7.3 Simulation Execution

Before executing the simulation, all agents and entities (i.e., product elements, activities, tasks, engineers, tools) are created based on input files that store the input data. The input data required for this is summarized in Table 7.1. The detailed data defined for the simulation can be found in Appendix C. Since no empirical data was available, the values for all

Table 7.1: Summary of the input parameters for the simulation model.

| Entity | Attributes | | |
|-------------------|-------------------------------|--------------------------------|--------------------------------------|
| <i>Product</i> | Elements Hierarchy | Interfaces Novelty | Knowledge Requirements Importance |
| <i>Activities</i> | Duration | Learning Rate | |
| <i>Engineers</i> | Responsibilities Expertise | Digital Literacy Experience | Salary |
| <i>Tools</i> | Capabilities Productivity | Usability Accuracy | Interoperability Cost |

parameters used are rough estimates based on observations from the literature.

The simulation is executed by stepping through time in small increments Δt , continuously assessing and updating the state of each agent at every step. State-transitions are triggered according to the logic described in the previous sections. The next section will describe what data is extracted during the simulation. After that, the efforts to calibrate the model will be discussed.

7.3.1 Data Extraction

In Section 3.3 many possible MOEs and MOPs were proposed to evaluate PD. Many of these are supported by the developed M&S framework. However, creating data extraction functions for all of the different states of agents and their interactions, as well as functions for the necessary visualization, is a time-consuming process. Therefore, in this thesis, only the data necessary to demonstrate the potential capabilities of the framework and validate the hypotheses will be extracted.

Because the simulation has many probabilistic effects (e.g., activity duration, validation success, information exchange, problem rates), a Monte Carlo simulation is needed to gain reliable insights. The simulation results are then averaged, and distributions or scatter plots can be utilized to visualize their spread.

Measures of Effectiveness

All MOEs (i.e., cost, time, quality, risk) are extracted during the simulation. However, quality is a fallout, meaning that a quality goal is predefined and the simulation runs until that goal is achieved through sufficient iteration. The development risk is calculated based on the cost and lead time distributions of the Monte Carlo simulation by applying Equation 3.1 and 3.2.

Measures of Performance

Some MOPs and other more detailed information is extracted during the simulation to measure how changes to the configuration impact the individual entities of the model. Table 7.2 lists the extracted MOPs along with the hypothesis each is intended to validate.

Measurements 1–4 track the state of agents and plot this over time, average it, or sum it up. These metrics are only available for single simulation runs. The cost tracked during simulation is derived from the salaries of engineers and the cost of using or licensing tools.

Measurements 5–10 have already been stated as potential MOPs in Section 3.3.2. Their basic definitions are unchanged, but some notes on their implementation for this simulation

Table 7.2: Summary of the measures of performance extracted during the simulation.

| MOP/Measurement | Parameter | Validation of |
|--|-----------------------|----------------|
| 1 Gantt chart ^a | – | |
| 2 Applied and backlogged effort over time ^a | – | Hypothesis 1 |
| 3 Resource utilization over time ^a | – | |
| 4 Effort and cost breakdown ^a | – | |
| 5 Rework effectiveness | η_{rework} | Hypothesis 2.1 |
| 6 Average number of iterations per element | $\overline{n_{iter}}$ | Hypothesis 2.1 |
| 7 First pass yield | FPY | Hypothesis 2.1 |
| 8 Cost of physical validation | $C_{physical}$ | Hypothesis 2.1 |
| 9 Work efficiency | η_{value} | Hypothesis 2.2 |
| 10 Average information consistency | $\overline{I_C}$ | Hypothesis 2.2 |

^a Only extracted for single simulation runs.

should be made: For metric No. six, the number of iterations of each element is counted and averaged over all elements of the product. The first pass yield (FPY) is the number of testing tasks that have not been repeated divided by the total number of testing tasks. Metric No. eight counts the cost of prototyping and testing activities as the cost of physical validation. Lastly, the average information consistency requires a detailed analysis of the information used by individual agents, which will be discussed in the following section.

Average Information Consistency

The information consistency I_C of individual agents, defined in Equation 7.26, is measured by comparing the information the agent uses while designing product elements with the available information. The consistency reaches 1 when no information is missing or outdated. Negative values are possible if the values for missing information and outdated information are large. The underlying assumption for this is that only missing information is better than missing the correct information and additionally using outdated information.

$$I_C = 1 - (I_m + I_o) \quad (7.26)$$

Missing information I_m is the difference between the available information and the amount of correct information that is used for a specific task, as stated in Equation 7.27. The available information I_a is the total information that has already been generated by other agents. The amount of correct information is defined through the completeness I_{comp} of the information, that is, the total amount of information used, and the correctness I_{corr} of that information. Outdated information I_o is the amount of incorrect information that is used for a specific task, as stated in Equation 7.28.

$$I_m = I_a - I_{comp} \cdot I_{corr} \quad (7.27)$$

$$I_o = I_{comp} \cdot (1 - I_{corr}) \quad (7.28)$$

The information consistency of an engineer is calculated for every dependent element after each design task is completed. After the simulation is completed, the average information consistency $\overline{I_C}$ is calculated across all agents, elements, and tasks.

7.3.2 Model Calibration and Verification

The model uses several tuning parameters and settings that can change the behavior and execution of the simulation that have been mentioned throughout this chapter. All parameters and their values are summarized in Table B.3 in Appendix B. Some initial calibration of these parameters was performed to obtain reasonably realistic results. However, since no empirical data was available, fine-tuning was not possible.

Verification of the intended behavior was performed during model development after each major update, ensuring that the defined logic is implemented correctly. This was done by tracing the states of all agents (product elements, engineers, activities, tools) and events that trigger state transitions in the simulation logs. In addition, several iterations of the completed framework were necessary on the basis of the simulation results to enhance the logic, correct errors, and handle edge cases.

The convergence and precision of the simulation was ensured by analyzing the influence of different step sizes Δt , task durations E_{td} , and number of simulation runs n_{runs} . A balanced approach was adopted to obtain sufficiently accurate data with stable distributions, while avoiding a significant increase in execution time. With the selected parameters, a full Monte Carlo simulation of one configuration is executed in 15 to 25 minutes on a high-end computer.

The next chapter will analyze the calibrated simulation framework to demonstrate and verify its capabilities, as well as to validate the hypotheses of this thesis based on the extracted data for different scenarios.

CHAPTER 8

RESULTS AND ANALYSIS

This chapter will test the developed simulation framework to validate or reject the hypotheses. For this, first the baseline configuration of the notional case study is simulated to evaluate the general simulation behavior and then test **Hypothesis 1**. After that, the sensitivity of the framework to various scenarios related to digital tools is evaluated, to test **Hypothesis 2.1** and **Hypothesis 2.2**. Additional capabilities of the framework will also be presented to demonstrate how the framework could help manage or plan product development, and therefore fulfill the **Research Objective**.

8.1 Analysis of Baseline Simulation Results

Before the impact of varying parameters through different scenarios can be analyzed, the baseline configuration must be simulated. The input data for this configuration is defined in Appendix C.

First, a Monte Carlo simulation of this configuration was performed to quantify its MOEs and MOPs. These will be used as a reference for the sensitivity analysis later in this chapter. Then a likely case is analyzed in more detail to highlight the framework's capability to provide deeper insights into causes of inefficiencies. This will also help with analyzing the behavior of the process in more detail to validate or reject **Hypothesis 1**.

8.1.1 Monte Carlo Simulation Results

The cost and lead time distributions resulting from the Monte Carlo simulation are depicted in Figure 8.1. The results have been normalized as the actual values do not provide any insight due to the use of a notional case study. All extracted MOEs and MOPs are summarized in Table 8.1.

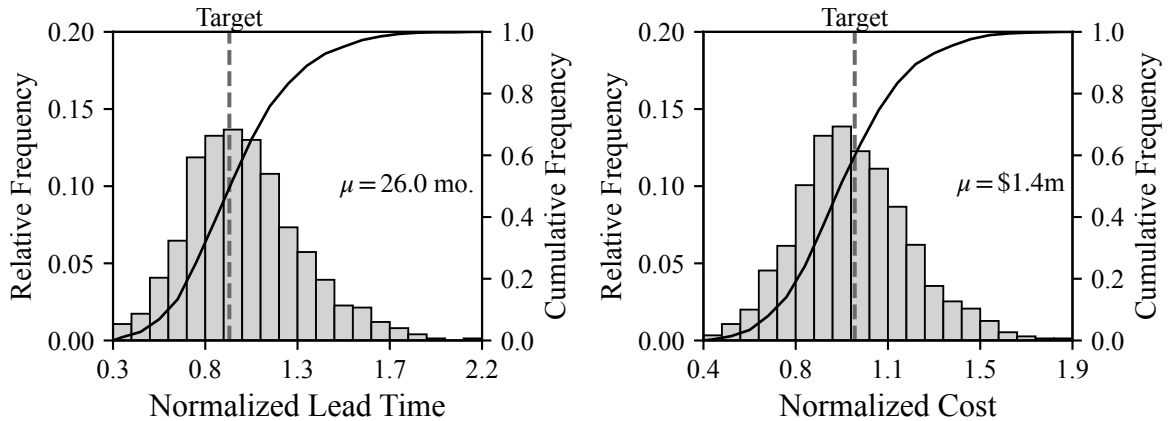


Figure 8.1: Lead time and cost distributions of the baseline configuration ($n_{runs} = 1,500$).

The spread of the distributions is large. This is because validation tasks are modeled probabilistically on the basis of the quality of architecture elements and do not incorporate decision logic. This leads to the possibility of no iteration occurring at low quality or excessive iteration occurring at high quality.

The distributions are used to calculate the risk (Equation 3.1 and 3.2), based on all runs that are above the cost and the lead time target multiplied by the impact of the overrun.

The quality is lower than the defined target quality of 0.9. Due to probabilistic quality checks, this is expected. When the quality is above the target, the probability of passing this check is 1. Therefore, on average, the quality will always be lower than the target. Since the simulation uses quality to converge, it is not indented as an evaluation metric and was only listed as a control.

Overall, the results show that the process heavily relies on iteration and rework. On average, every element is reworked 7.5 times, resulting in only 27.5% of development work not being reworked. Much of this rework is caused by physical validation as only 7.5% of physical validation tasks are successful on the first pass and 72% of the development cost is caused by physical validation. In addition, only 49% of the development effort is spent on value-adding work, i.e. technical tasks, and the average consistency of the information used by engineers is 66%, which means that on average 34% of the information is missing or outdated while designing individual components.

Table 8.1: MOEs and MOPs of the baseline configuration ($n_{runs} = 1500$).

| Metric | Definition | Result \pm Confidence (95%) |
|-----------------------|--------------------------------------|-------------------------------|
| Cost | Cost of labor and tools | \$1.40m \pm \$0.02m |
| Lead Time | Time for completion | 26.0 mo. \pm 0.3mo. |
| Risk | Eq. 3.1 and Eq. 3.2 | \$18.76m |
| Quality | Overall Solution Goodness (Eq. 7.13) | 0.804 \pm 0.006 |
| η_{rework} | Rework Effectiveness (Eq. 3.4) | 0.275 \pm 0.002 |
| $\overline{n_{iter}}$ | Average number of iterations | 7.5 \pm 0.1 |
| FPY | First pass yield of testing tasks | 0.075 \pm 0.002 |
| $C_{physical}$ | Relative cost of physical validation | 0.720 \pm 0.002 |
| η_{value} | Work Efficiency (Eq. 3.3) | 0.490 \pm 0.002 |
| $\overline{I_C}$ | Information Consistency (Sec. 7.3.1) | 0.660 \pm 0.001 |

8.1.2 Dashboard for Detailed Insights

A simple dashboard was created to visualize the data extracted during a single simulation run. The data relates to measurements 1–4 in Table 7.2. Figure 8.2 presents the dashboard and the extracted data from a likely run of the baseline configuration. The run was selected based on its close proximity to the mode of the cost and lead time distributions.

The left-hand side of the dashboard displays time-series data about the process and engineers. The upper diagram shows a Gantt chart generated by the simulation. Blue bars indicate working on a task for the first time, red bars indicate repetitions of tasks due to rework, and gray bars indicate paused tasks due to technical problems, missing information, or resource constraints. The center diagram plots the effort backlog of different teams, meaning the effort of tasks that are ready to be performed. The bottom diagram displays the resource utilization of the same teams.

The right-hand side displays the effort and cost breakdown of the completed process. The upper diagram breaks down the effort of individual teams spent on different types of activities such as technical work, collaboration, information handling, and waiting (i.e. for information or collaboration). The other diagrams break down the cost of the components

(center) as well as the product and subsystems (bottom) by different types of activities. In this case Development (i.e. System Design, Design, Virtual Integration), Virtual Validation, and Physical Validation are the selected types.

Identification of Inefficiencies

The dashboard can be used to analyze the causes of inefficiency and how the process can be improved. The times-series data (left-hand side) can be used to identify bottlenecks due to under-staffing. When multiple activities are assigned to the same engineer or team, but cannot be processed, the effort backlog rises, while the resource utilization is constant. In addition, paused tasks on the Gantt chart indicate resource constraints as well. In the example, it can be seen that the Testing and Prototyping Team is a bottleneck for the process because it has high peaks in the effort backlog and many paused tasks in the Gantt chart.

The cost breakdown can help identify what activities or parts of the product cause the most cost. The reason for this could be high complexity, but also missing competencies or predictive capabilities.

Breaking down effort allows for a clearer understanding of time losses and their causes. In the example, the Design and Electronics Team spends relatively much time waiting for information, causing them to stop working. Here, measures such as increasing the interoperability could be identified to mitigate these inefficiencies.

In general, the analysis of the data presented in this dashboard offers detailed insight into the specific sources of inefficiencies, rather than simply reporting high-level metrics related to overall effectiveness and performance. This enables a more targeted identification of areas that need or could benefit from improvement.

More detail could be added, by extracting more data and enabling dynamic visualizations that allow the user to quickly filter data or step through time. Using data from the Monte Carlo simulation instead of the most likely run could increase accuracy. This would require averaged time-series data to be visualized probabilistically, as proposed by [105].

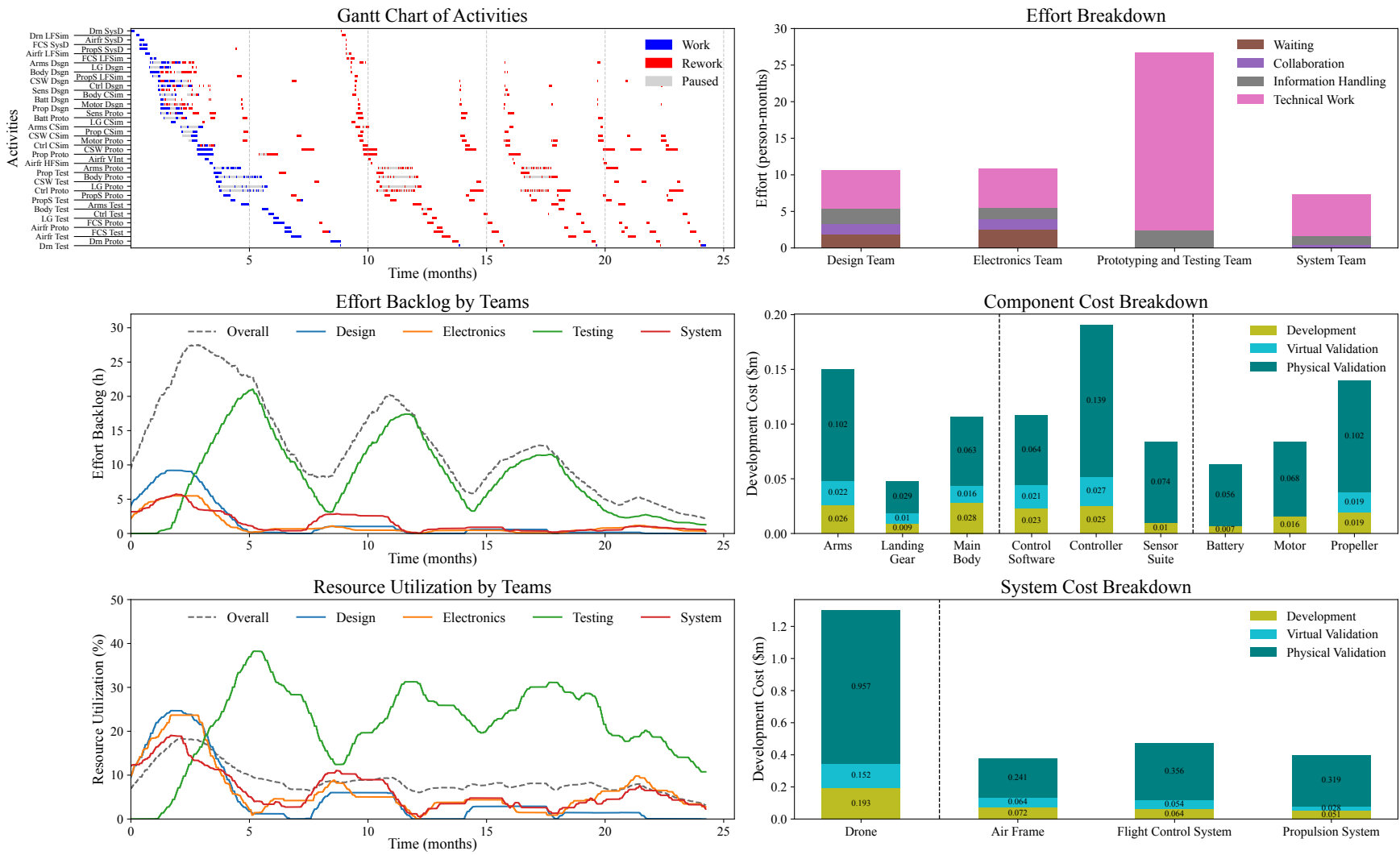


Figure 8.2: Dashboard visualizing data extracted during a single simulation run of a likely run of the baseline configuration (Effort backlog and resource utilization have been smoothed using a moving average).

8.1.3 Analysis of the Process Behavior

Based on the most likely run of the Monte Carlo simulation, the behavior of the process will be analyzed in more detail, to test **Hypothesis 1**: *If generic process elements are used to recursively generate activities based on the product architecture, then a product development process can be simulated more accurately without significantly increasing the modeling effort.*

Figure 8.3 plots the effort applied by engineers over time for development and testing activities, as well as the overall process. Here, the highest peaks are caused by the initial design and testing activities. During rework, the most effort is caused by testing activities as these activities have low learning rates and have to be repeated fully to validate a new design. This can also be seen in the Gantt chart of Figure 8.2. The large amount of rework is caused by several iterative loops that can span the entire process. For smaller scale projects this might be possible, however, for large engineering projects gates are usually implemented to limit iteration between stages that can cause this kind of large-scale rework [38]. The effects of gates or reviews, such as the preliminary design review (PDR) or critical design review (CDR), are not considered in the simulation framework.

Compared to the baseline model (Figure 8.4) the effort distribution is shifted towards Norden's effort model [129], meaning that the process and the interactions during the pro-

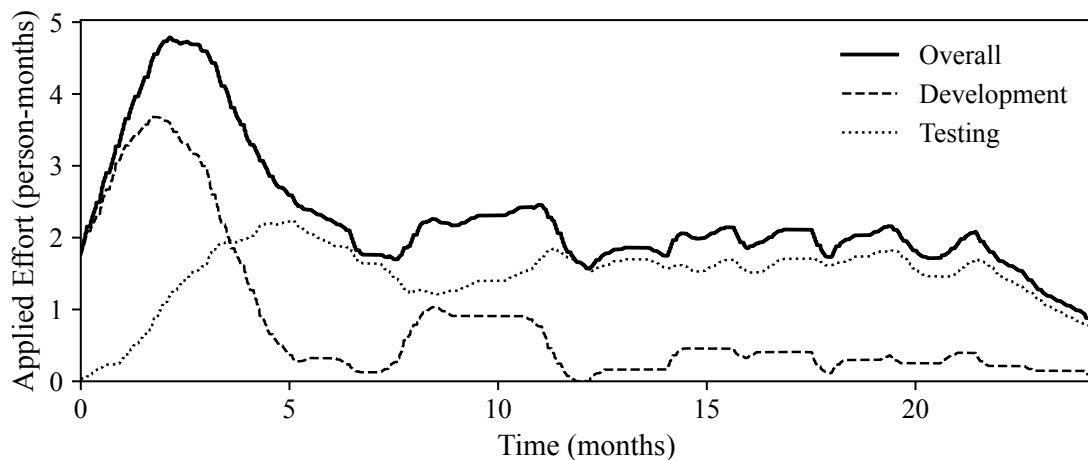


Figure 8.3: Applied effort over time of a likely run of the baseline configuration. Results have been smoothed using a moving average.

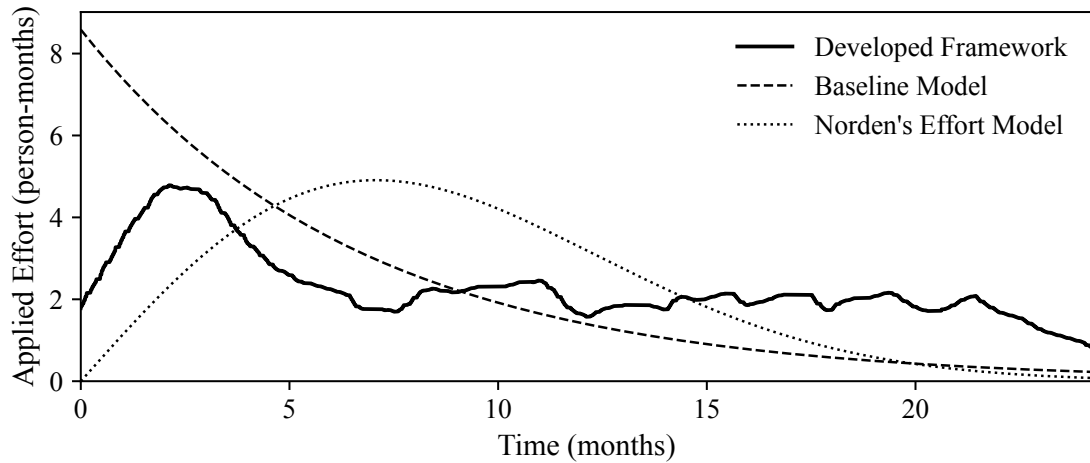


Figure 8.4: Comparison of the effort distributions generated by the developed simulation framework, the baseline model (approximated) and Norden’s effort model (Equation 5.1).

cess are more realistic. In particular, the beginning of the process is improved by explicitly including system design activities. This leads to the required effort increasing first before decreasing, instead of starting high and continuously decreasing. As a result, the normalized mean squared error is reduced from 0.65 to 0.35. Therefore, the developed simulation framework improves the results despite the excessive amount of rework that increases the effort towards the end of the process.

Furthermore, the modeling effort to use this simulation framework is comparable to the modeling effort of the baseline model. Individual duration distributions for the different types of activities (Table C.2 in Appendix C) are the only additional inputs needed. Therefore, the ability to simulate the PD process more realistically than the baseline model while maintaining the modeling effort leads to the validation of **Hypothesis 1**.

Although the process is simulated more realistically, some improvements are possible to move closer towards the effort distribution proposed by Norden’s effort model. The spread of the simulation results is high because the validation tasks are modeled probabilistically. This can cause large-scale rework that might not be realistic for PD projects that usually use gated processes. Stochasticity could be reduced by implementing review gates (PDR, CDR) and adding decision logic that defines a minimum quality requirement that always

has to be met but also limits the amount of rework. Limiting the amount of rework can be done by adapting the risk calculation to include the quality. The simulation then converges by assessing the risk of iteration, weighing the added time and cost against its potential quality gains. This approach was already proposed by [115] for activity networks.

8.2 Sensitivity Analysis

To test the impact of digital tools using the proposed M&S framework, the variables related to these tools will be analyzed. This will enable the validation or rejection of

Hypothesis 2.1: *Digital tools can be modeled as agents that are used by agents of engineers for the execution of activities, where the efficiency and quality of results depend upon the tool's properties and the engineer's digital literacy; and*

Hypothesis 2.2: *Collaborative and knowledge management tools can be modeled as central platforms that connect tools, personnel, and knowledge with each other with varying efficiency depending on interoperability and digital literacy.*

Several Design of Experiments (DoE) were created to analyze different aspects of the modeled tools related to these hypotheses. First, smaller DoEs will investigate individual effects related to accuracy, interoperability, and digital literacy, as well as the addition of a new tool. After that, a larger DoE will investigate the design space of the organization by combining the parameters of the previous DoEs and increasing the granularity. The following sections will describe the experiments in more detail and present their results.

8.2.1 Experiment 1 — Accuracy of Simulation Tools

This experiment focuses on the impact of the Engineering tools on the validation of designs. The accuracy of the digital tools is varied by a percentage relative to the baseline configuration's values. In addition, the digital literacy of engineers will be varied to test the impact of training. All agents and tools will be changed at once for this experiment. Table 8.2 summarizes the setup of this DoE.

Table 8.2: Setup of Experiment 1.

| Parameters | | Range | Design | Cases | Runs per case |
|---------------------|--------------|------------|----------------|-------|---------------|
| Digital literacy | $A_{DL,Eng}$ | {1, 2, 3} | Full factorial | 27 | 400 |
| Factor for accuracy | T_{Acc} | -0.8...0.8 | | | |

Simulation Results

Figure 8.5 summarizes the MOEs and MOPs extracted during all the cases simulated for Experiment 1. The mean and confidence of each case are plotted. Cost, lead time, and risk have been normalized around the mean values generated by the baseline simulation. The baseline value is indicated for all metrics to better visualize the relative changes.

The overall cost (a) and lead time (b), and consequently risk (c), are reduced by increasing the accuracy and digital literacy. Digital literacy has a strong negative effect on the metrics when it is lower than required by the tool used. The opposite effect is weaker because less efficiency gains and no accuracy gains are possible for excessive expertise. The risk plot (c) has some outliers, probably caused by the variability of the results. Because risk is calculated based on the cost and lead time distributions (Eq. 3.1 and 3.2) it is particularly sensitive to slight instabilities. Quality (d) is unaffected by all changes as it is used for the convergence of the simulation.

As a result of the increased predictive capability, increasing the accuracy parameter leads to less rework being required, thus increasing the effectiveness (e). At the same time, the average number of iterations (f) slightly increases, indicating that while more iteration is caused by better virtual validation tools, it occurs earlier and therefore is more efficient. These effects can also be seen in the first pass yield (g) and relative cost of physical validation (h).

Work efficiency (i) is negatively affected by the digital literacy of engineering tools, because technical work is performed faster, but supporting work is unchanged. Information consistency (j) is not affected by changing the accuracy of the tools. This is expected because the exchange and processing of information is not influenced by a tool's accuracy.

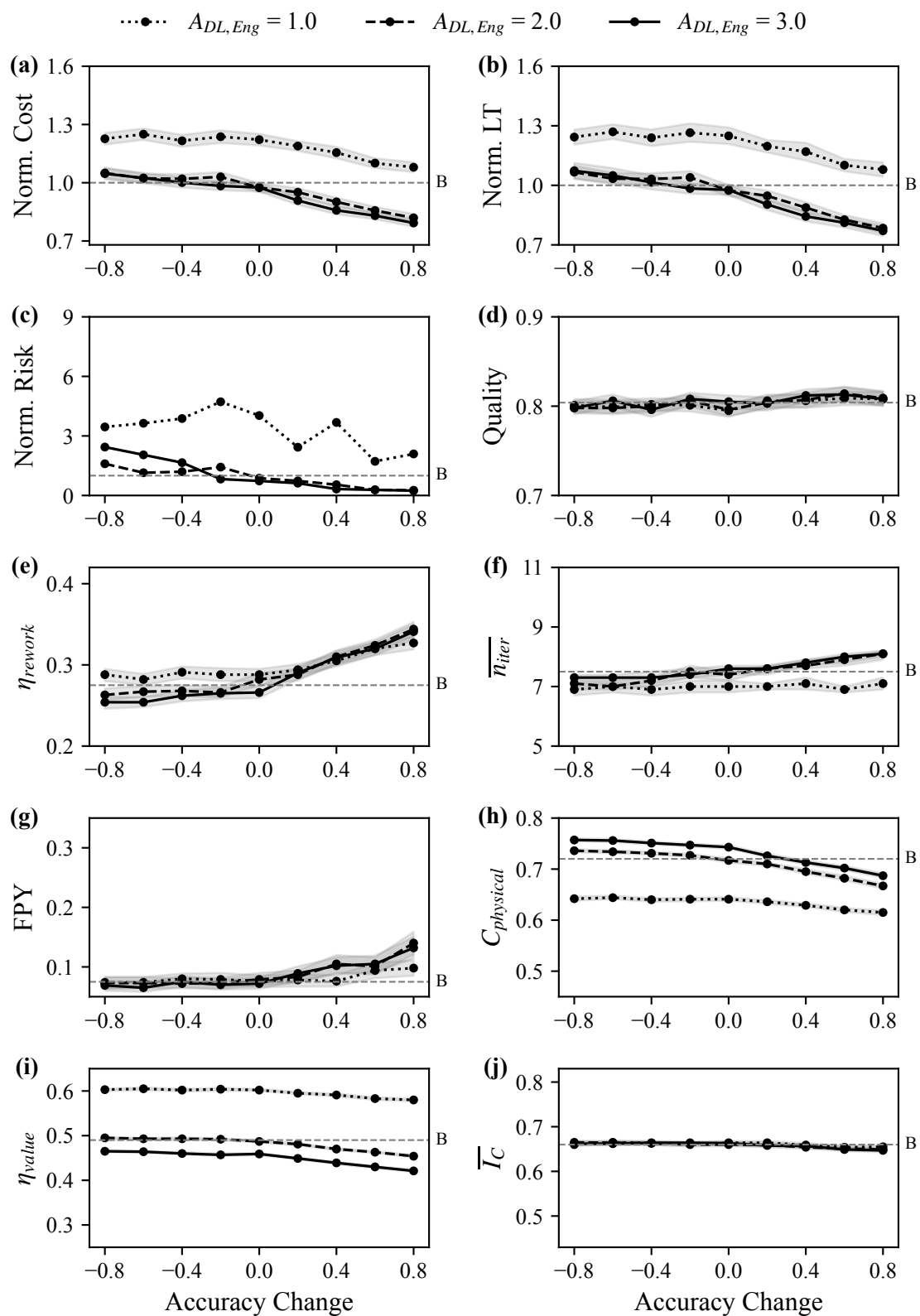


Figure 8.5: Results of Experiment 1 — Accuracy and digital literacy (Gray area: 95% confidence interval; B: Result of the baseline configuration).

8.2.2 Experiment 2 — Interoperability between Tools

This experiment focuses on the impact of the EKM tool and its interaction with the Engineering tools on the exchange of information during simulation. For this, the interoperability of tools and the digital literacy of engineers are varied. All agents and tools will be changed at once for this experiment. Table 8.3 summarizes the setup of this DoE.

Table 8.3: Setup of Experiment 2.

| Parameters | | Range | Design | Cases | Runs per case |
|------------------|--------------|-----------|----------------|-------|---------------|
| Digital literacy | $A_{DL,EKM}$ | {1, 2, 3} | Full factorial | 33 | 400 |
| Interoperability | T_I | 0...1 | | | |

Simulation Results

Figure 8.6 summarizes the MOEs and MOPs extracted during all the cases simulated for Experiment 2. Cost (a), lead time (b), and risk (c) are reduced due to increased interoperability and digital literacy. Interestingly, the effect is stronger at lower levels of interoperability, suggesting that initial improvements in interoperability yield greater gains than later ones. Quality (d) is again not affected by the changes.

These improvements result from reduced effort in processing information, which enables faster and more frequent information exchange. Therefore, the work efficiency (i) and the average information consistency (j) are increased. For high interoperability, the impact of digital literacy on the work efficiency is reduced due to higher degrees of automation of information processing tasks. The inverse is observed for the information consistency because the effective use and exchange of information still depends on the competence of the engineer.

As a result of the use of more consistent information during design activities, the quality of interfaces is higher. This makes iteration (f) less necessary to fix interfaces and increases the effectiveness of process (e) by reducing the amount of repeated work. The

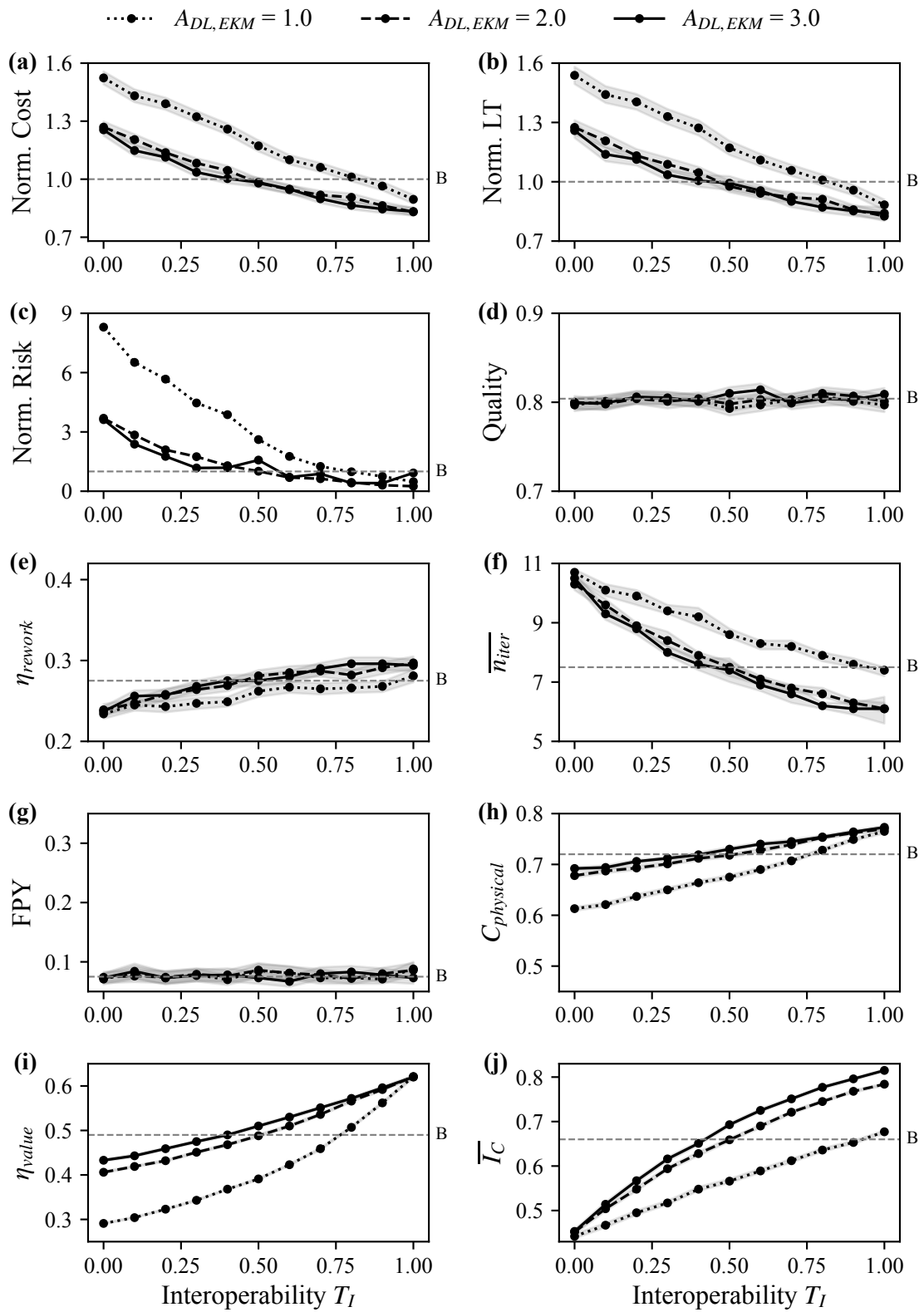


Figure 8.6: Results of Experiment 2 — Interoperability and digital literacy (Gray area: 95% confidence interval; B: Result of the baseline configuration).

first pass yield (g) remains unchanged due to design quality issues not being affected by information exchange. In addition, it is unaffected because the verification of the interface by the system designers takes place before physical testing begins.

Unintuitively, the relative cost of physical validation (h) increases with better interoperability. This is because the reduction of cost for information processing is greater than the cost saved due to reduced physical testing needs.

8.2.3 Experiment 3 — New Tool Capability

In this experiment, a new tool is added to the organization. This “Advanced System Simulation Tool” enables activities related to *Virtual Integration* and *High-fidelity System Simulation* of the *Propulsion System*, *Flight Control System*, and *Drone*. Due to the logic of the process generation, discussed in Section 6.2, new activities are created to account for the capabilities that the tool adds.

Figure 8.7 and Figure 8.8 compare the activity networks of the baseline configuration and the new configuration with the added tool. The structure of the process changes substantially. According to the definition of the high-level process used for this thesis (Figure 6.2), physical validation activities only begin once all elements that are hierarchically dependent have been validated virtually. Thus, with the newly added tool, all prototyping activities start after the *HF System Simulation* activity of the Drone is completed.

To assess the impact of the new tool and identify the characteristics necessary for it to enhance the PD process, its parameters are analyzed. These are the accuracy, usability, and interoperability of the tool. Table 8.4 summarizes the setup of this DoE.

Table 8.4: Setup of Experiment 3.

| Parameters | | Range | Design | Cases | Runs per case |
|------------------|------------|-----------------|----------------|-------|---------------|
| Usability | T_{Usab} | {1, 2, 3} | | | |
| Accuracy | T_{Acc} | 0.1...0.9 | Full factorial | 81 | 400 |
| Interoperability | T_I | {0.2, 0.5, 0.8} | | | |

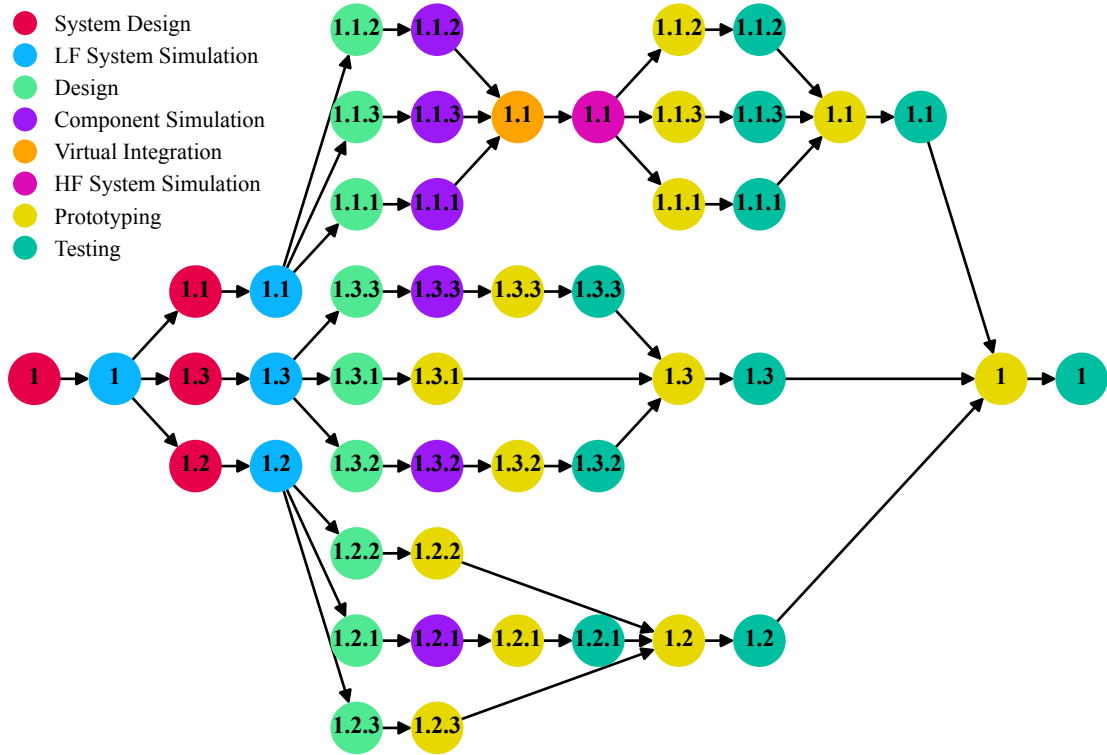


Figure 8.7: Activity network generated by the baseline configuration.

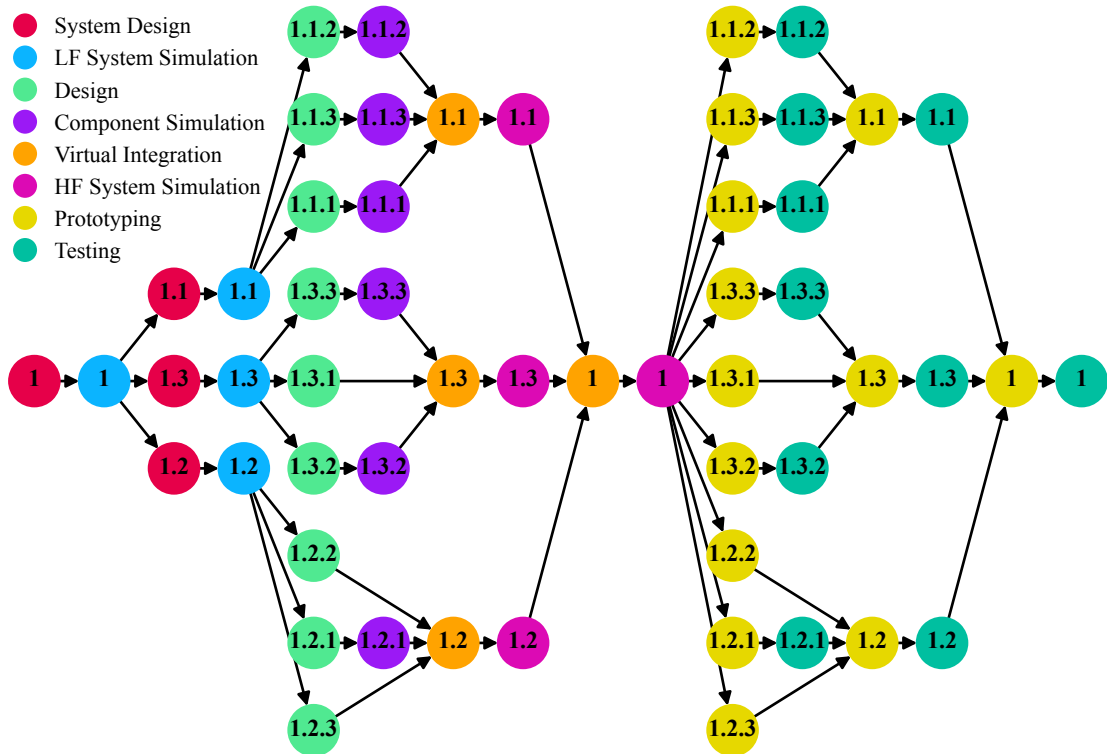


Figure 8.8: Activity network generated by adding the new HF system simulation tool.

Simulation Results

The results of this experiment are summarized for the tool’s interoperability $T_I = 0.5$ in Figure 8.10. Detailed results for changes of the interoperability are not provided as these only slightly impact the simulation results, as illustrated for a few examples in Figure 8.9. The reason for this is that for simulation activities, information is processed only once at the beginning and end.

The impact of the tool’s accuracy is similar to the results of Experiment 1. However, performance improvements are only made above a certain level of accuracy. For this configuration, the tipping point is $T_{Acc} = 0.5$. The new tool is capable of substantially reducing the need for physical testing (g, h) beyond the tipping point. As a result, the effectiveness (e) is increased. Rework is still required, as the tool does not improve quality directly, but quality issues are detected earlier. However, the number of iterations (f) also increases slightly. This shows that a highly capable tool allows for more design iterations while applying only some additional effort. The effects observed here are stronger than with the accuracy of the tools tested in Experiment 1 because the high-fidelity system simulation tool can virtually validate the fully integrated system, which includes all its interfaces.

For low usability (i.e., large values for T_{Usab}), no performance improvements are made at any level of accuracy, as higher levels of digital literacy are required of the engineer using

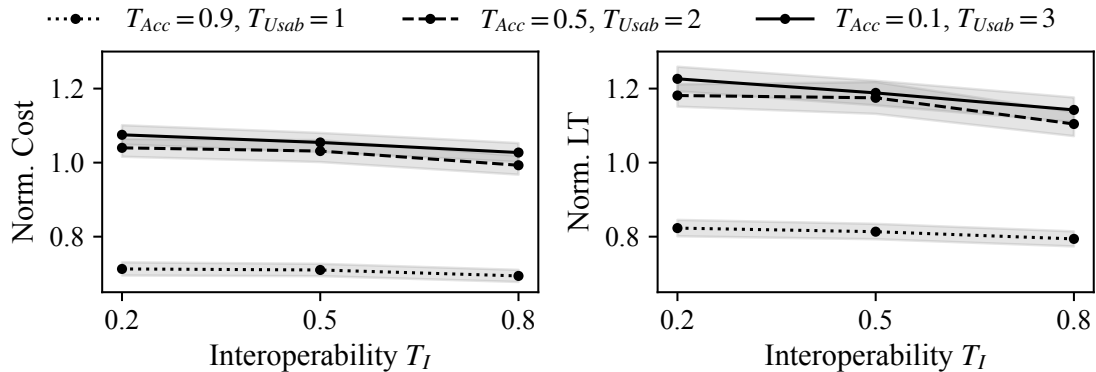


Figure 8.9: Results of Experiment 3 — Interoperability of the new tool (Gray area: 95% confidence interval).

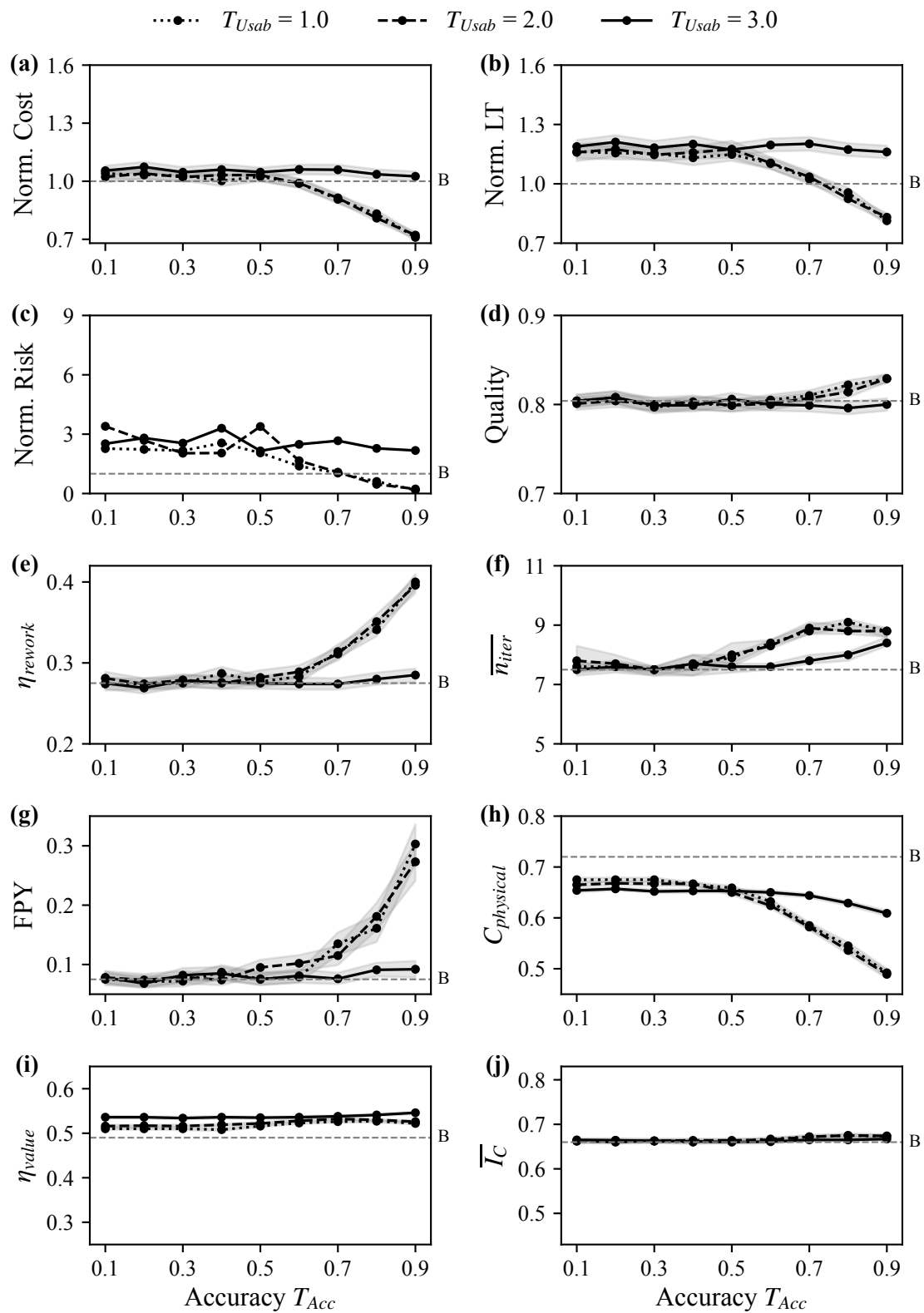


Figure 8.10: Results of Experiment 3 — Accuracy and usability of the new tool (Interoperability $T_I = 0.5$; Gray area: 95% confidence interval; B: Result of the baseline configuration).

the tool. Notably, the overall PD performance even is worse than the baseline configuration due to increased cost (a) and lead time (b). This is caused by the extra effort required for the newly added activities and potentially new resource constraints due to changes in the process structure. As illustrated in Figure 8.8, all physical validation activities begin after the virtual validation is completed. This creates new bottlenecks that were not as extreme in the structure of the baseline process.

To outweigh these added challenges, the accuracy of the tool and the digital literacy of the engineer who uses the new tool must be sufficiently high. If this is not the case, the activities enabled by the new tool do not add value to the PD process, due to most quality issues remaining undetected. Only when the tool has both adequate accuracy and good usability the additional effort pays off, as it enables earlier and faster design iteration.

8.2.4 Experiment 4 — Exploration of the Organizational Design Space

To explore the design space of the organization in more detail, a DoE is performed that individually varies the parameters of each tool. The selected parameters are the same as in Experiments 1–3. However, because of the high dimensionality of this experiment, a full factorial DoE is not feasible. A design consisting of a D-optimal design generated by JMP

Table 8.5: Overview of Experiment 4.

| Parameters | | Range | Design | Cases (Runs per case) |
|--------------------------------|--------------|-----------------|-----------|-----------------------|
| Digital literacy | $A_{DL,Eng}$ | {1, 2, 3} | | |
| Digital literacy | $A_{DL,EKM}$ | {1, 2, 3} | | |
| <i>Each tool individually:</i> | | | | |
| Accuracy | T_{Acc} | {0.2, 0, 0.8} | D-optimal | 276 (400) |
| Interoperability | T_I | {0.2, 0.5, 0.8} | + | + |
| <i>New tool:</i> | | {Yes, No} | LHS | 724 (50) |
| Usability | T_{Usab} | {1, 2, 3} | | |
| Accuracy | T_{Acc} | {1, 2, 3} | | |
| Interoperability | T_I | {0.2, 0.5, 0.8} | | |

[138] and Latin Hypercube Sampling (LHS) [139] was chosen. This particular D-optimal design is similar to a Box-Behnken Design [140], as it uses three levels and avoids corner points. The LHS is applied to the full range of possible values. This combined approach ensures that the edges of the design space are included, but also that the space in between is filled. A total of 1,000 cases were simulated. Table 8.5 summarizes the setup of this DoE.

Simulation Results

Figure 8.11 and 8.12 aggregate all results of Experiment 4 in matrices related to the accuracy and interoperability of the tools, respectively. The x-axis represents the input variables, while the y-axis shows the responses for all MOEs and MOPs. Given the large volume of data and its variance, only the linear regression lines are plotted to help illustrate the strength and direction of the effects more clearly. Regression was performed on data from individual simulation runs. Since the addition of the new tool is a categorical variable, with its properties only relevant when set to ‘true’, the results are plotted separately for cases where the tool is used versus when it is not.

It can be seen that the strongest effects are digital literacy (EKM and Engineering), the accuracy of the MBSE tool, and the addition of the HF system simulation tool. The overall impact of the digital literacy is high because it is applied to all agents of engineers at once and affects the efficient use of all tools.

The accuracy of the MBSE tool (i.e., system model fidelity, completeness) not only impacts the analysis of possible feasibility issues in system designs, but is also used to perform design verifications. Improved accuracy improves the detection of interface compatibility issues, minimizing the need for unnecessary rework.

The main effects of the new tool are the same as stated in Experiment 3 (Section 8.2.3), but additional effects can also be identified. For example, the accuracy of component simulation tools (i.e., FEM, Circuit Simulation) has a greater impact when the new HF system simulation tool is added. A possible cause for this is that the new tool enables faster it-

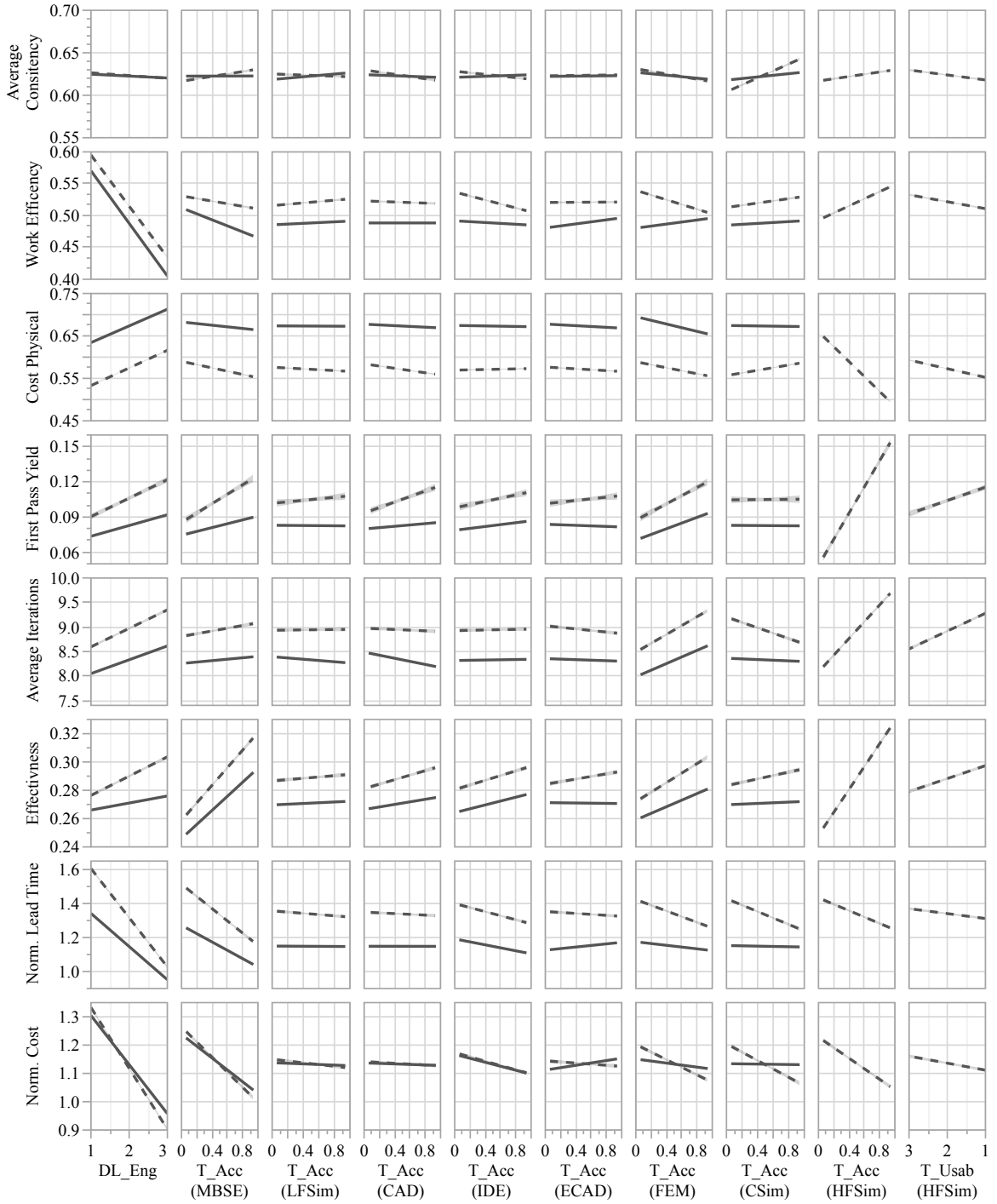


Figure 8.11: Linear regression results of Experiment 4 — Accuracy (dashed: with new tool).

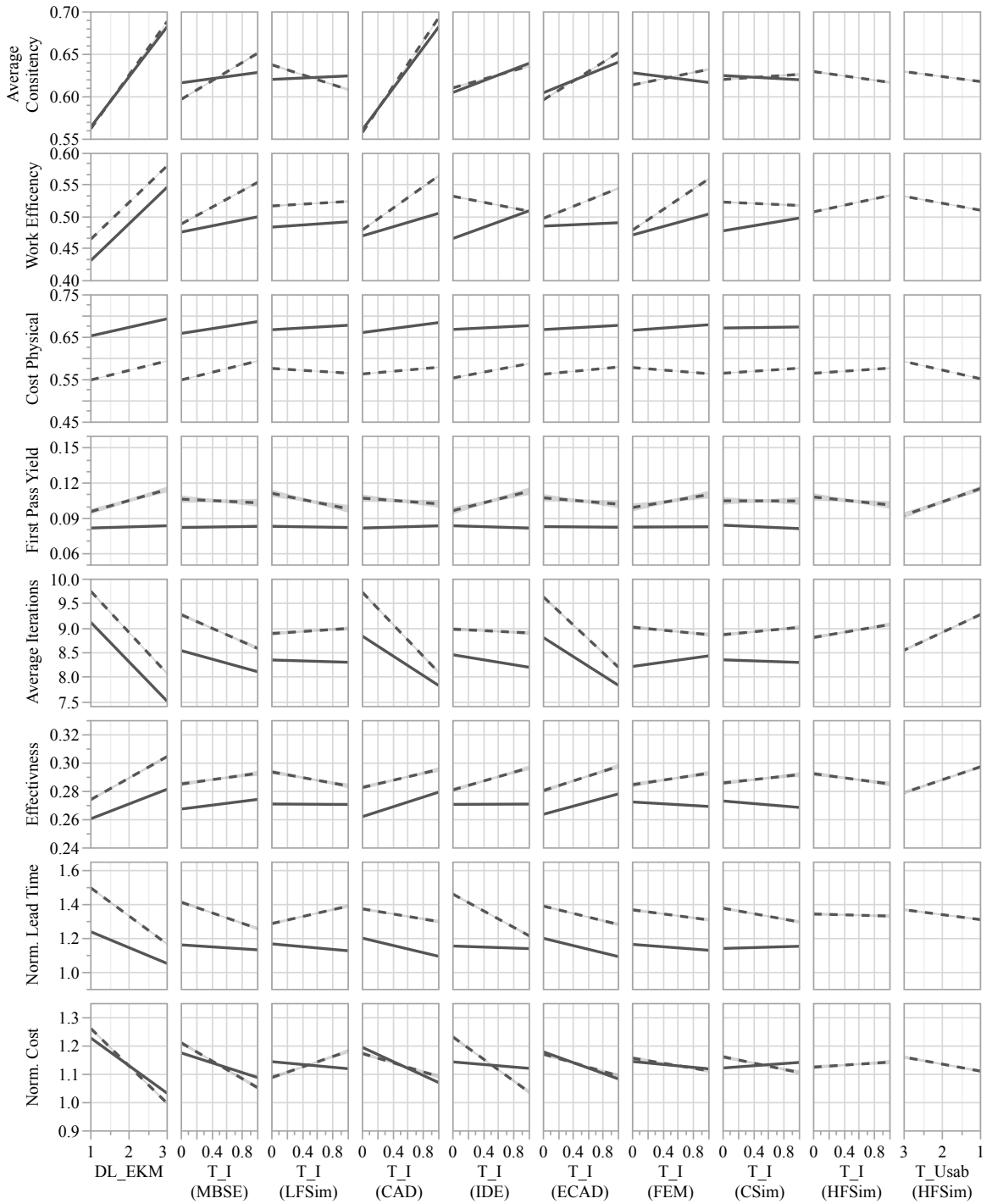


Figure 8.12: Linear regression results of Experiment 4 — Interoperability (dashed: with new tool).

eration which is also beneficial for the component simulation tool because many different designs can be tested.

Design tools, such as CAD and ECAD, also have an accuracy parameter that allows these tools to assist in identifying feasibility issues. This represents the integration of simulation or evaluation capabilities into design tools. However, the impact of these capabilities is not significant in the simulation, likely due to interface and design quality issues outweighing the feasibility.

Analyzing the interoperability of individual tools reveals that some benefit more from higher interoperability than others, primarily due to their need to exchange information. Among these, the MBSE and CAD tools have the greatest impact on the results. Since the CAD tool is responsible for developing many of the defined product elements, it also plays a key role in shaping their interfaces, making the exchange of data crucial. Meanwhile, the MBSE tool receives information in order to perform design verifications.

Further Analysis of the Design Space

Analyzing the interactions between more tools, similar to the current analysis of the new tool, would allow one to identify more effects caused by multiple tools. In addition, interactions and correlations between different input parameters or metrics could be investigated. The large amount of data and metrics extracted during the simulation allows for many different types of analysis, depending on the need and interest of a potential user. However, this will not be further investigated as part of this thesis, as it would require the analysis of more granular data. Granular data can be generated, but currently provides limited insights due to its large variance (individual runs) or is very time-intensive to obtain (Monte Carlo).

Nevertheless, the previously presented results show that the design space can be analyzed to identify the tools that impact the PD process the most. In addition, interactions between different factors can be analyzed to better understand the behavior of the PD process. These insights could then, for example, support better decision making on what tools

to implement or improve. Also, the number of configurations to be considered for further evaluations or implementation can be reduced based on MOE/MOP requirements or goals.

8.2.5 Summary of the Effects of Digital Tools

The sensitivity analysis has revealed that the implemented agents of digital tools have a significant impact on the simulated PD process. Most of the observed effects are consistent with expectations. Some minor discrepancies are present, likely due to the variability in the simulation. Running additional simulations would help reduce these errors. Further fine-tuning the strength of some effects could also reduce some unintuitive behaviors. The impact of digital tools can be summarized as follows.

Increasing the *Accuracy* of digital tools leads to faster and more cost-effective PD, due to more effective virtual validation and less large-scale rework. Consequently, the reliance on physical tests is also reduced. These effects are especially strong for the MBSE tool (i.e., system model fidelity, completeness) and high-fidelity system simulation tools (i.e., predictive capability about the integrated systems). The impact that accuracy has is weakened or completely negated when the digital literacy of engineers is not sufficient for the tool used. Therefore, **Hypothesis 2.1**, stating that *digital tools can be modeled as agents that are used by agents of engineers for the execution of activities, where the efficiency and quality of results depend upon the tool's properties and the engineer's digital literacy*, is validated.

Higher *Interoperability* leads to faster and more frequent exchange of information. This increases the consistency of the information used to design components, causing less rework. This effect is also significantly influenced by the digital literacy of the engineer using the tool. Interoperability is a more important factor for tools that require frequent information exchange, such as design tools and verification tools. These observations lead to the validation of **Hypothesis 2.2**, stating that *collaborative and knowledge management tools can be modeled as central platforms that connect tools, personnel, and knowledge*

with each other with varying efficiency depending on interoperability and digital literacy.

Although the direction of many observed effects is intuitive through the definition of the parameters, the strength of the effects is not and depends on many factors defined by the scenario (i.e., product, engineers, and other tools). In addition, the synergetic impact of various tools might not always be apparent, such as the impact of adding a new tool on the existing tools. The simulation can identify and quantify these interactions and effects to enable a more detailed analysis of which tools can benefit PD the most.

The simulation currently does not account for the cost and effort (i.e., investments into infrastructure, development of new tools/capabilities, training, etc.) required to implement changes. In order to conduct trade studies, the cost of implementation must also be taken into account. Therefore, the results must always be weighted against the cost of implementing the simulated configuration. For example, the interoperability of the CAD and MBSE tools both have similar responses in Figure 8.12, but increasing the interoperability of CAD tools will probably be easier than for MBSE tools [19].

CHAPTER 9

CONCLUSION

Product development is a difficult-to-manage process that is constantly changing and must adapt to new challenges. The increasing complexity of products and shift towards digital engineering change the engineering process itself. However, the planning and management processes of PD must also account for these changes. Therefore, the objective of this thesis was to *create a holistic modeling and simulation framework of product development that can serve as a decision support tool for managers to identify how new configurations or the implementation of new digital tools can lead to better product development performance.*

Several gaps of existing simulation methods were derived based on a set of observations on the characteristics of PD. The method that best met the defined evaluation criteria was selected as the baseline model for this thesis. The baseline model [106] was improved by adding new logic and entities to more holistically represent product development. The main additions consisted of a refined high-level process and digital tools used for engineering and knowledge management activities.

9.1 Review of the Hypotheses

Based on the identified gaps, additional logic was incorporated into the selected baseline model. These extensions were formulated as the following hypotheses.

Hypothesis 1: *If generic process elements are used to recursively generate activities based on the product architecture, then a product development process can be simulated more accurately without significantly increasing the modeling effort.*

Hypothesis 2.1: *Digital tools can be modeled as agents that are used by agents of engineers for the execution of activities, where the efficiency and quality of results depend upon the tool's properties and the engineer's digital literacy.*

Hypothesis 2.2: *Collaborative and knowledge management tools can be modeled as central platforms that connect tools, personnel, and knowledge with each other with varying efficiency depending on interoperability and digital literacy.*

A notional case study was used to demonstrate the capabilities of the framework and test the hypotheses. **Hypothesis 1** was tested by analyzing the behavior of the process. The analysis was focused on proving that using a generic high-level PD process for the simulation generates a more realistically distributed effort-time curve than the baseline model. For this, Norden's effort model [129] was referenced as a realistic effort distribution. The simulation results were not able to perfectly replicate Norden's effort model, but the effort distribution was shifted toward it compared to the baseline model and significantly reduced the normalized mean squared error from 0.65 to 0.35.

Furthermore, the main input for creating the process is the product architecture for both the baseline model and the model of this thesis. The modeling effort therefore remains unchanged. These results led to the validation of **Hypothesis 1**.

For **Hypotheses 2.1** and **2.2** detailed sensitivity analyses were performed based on various experiments. The main parameters (i.e., accuracy, interoperability, digital literacy) of digital tools were extensively studied. The results revealed that the modeled agents of the tools have significant effects on nearly all MOEs and MOPs used to evaluate the PD process. Increasing interoperability leads to more consistent information and less time wasted on non-value-adding activities. Higher accuracy reduces the reliance on physical testing and leads to less rework. Furthermore, the 'competency-capability' trade-off related to the digital literacy can be observed in the simulation results.

These effects are consistent with the expected behavior, but also go beyond it, by quantifying the strength of the effects and discovering interactions between tools. A deeper analysis discovered that some tools impact the results more than others depending on how many product elements are developed using a tool and what types of activities the tool is used for. These results led to the validation of **Hypotheses 2.1** and **2.2**.

9.2 Review of the Simulation Framework

This thesis was based on the gap in holistic views in PD simulations, as well as the lack of capabilities to quantitatively evaluate the PD process in a detailed and insightful way. Therefore this sections re-examines these gaps to identify the contributions of this thesis.

9.2.1 Modeling Content

The simulation combines all five domains of product development (product, process, organization, tools, goals) into a single quantitative evaluation tool. In Chapter 4 existing methods were evaluated based on the characteristics of PD. Table 9.1 provides the same evaluation, but also includes the simulation model developed in this thesis. The developed framework extends the *Knowledge-based Team Interaction* model [106] by adding a more detailed PD process and agents of digital tools.

The complexity of the product is included in the model to the same extent as the baseline model. The main complexity drivers, such as hierarchy, interfaces, and multidisciplinary, are considered.

The newly introduced high-level process within the simulation logic, which incorporates various development and validation activities, results in a more intricate and realistic representation of a PD process. However, the process is still relatively rigid, which leads to limited possibilities for changing the process sequence or overlapping activities. Formal reviews and gates (i.e., PDR, CDR), which are usually part of PD processes [29], are also not yet included in the model, leading to the possibility of very large rework loops.

The simulation model considers rework of activities that have already been completed to increase the quality of the product. This is triggered by validation activities (i.e., simulation, testing) or the review of designs (interfaces, feasibility). Rework propagates through the process sequence, causing additional rework for downstream activities (first-order rework). However, the propagation of changes over interfaces (second-order rework), which

Table 9.1: Comparison of the developed M&S framework with the existing approaches.

| M&S Approach | Modeling Content for a Holistic Representation | | | | | | | |
|---|--|--------------------------|-----------------------------|------------------------------|-----------|-------------------|-----------------------------|---------------------|
| | Product Complexity | High Level SE Process | Process Interactions | Organization & Communication | Knowledge | Tool Capabilities | Collaboration through Tools | Individual Entities |
| DES — Activity Network | ○ | ● | ● | ○ | ○* | ○* | | ○* |
| SD — Rework Loop | ○ | ◐* | ○ | | ○ | | | |
| ABS — Design Cognition | ● | ○* | ○ | ● | ○ | | | ◐ |
| ABS — Team Interaction (A) | ○ | ○ | ◐ | ● | ○ | ○* | ○* | ● |
| ABS — Team Interaction (K)* | ● | ○ | ○ | ◐ | ● | | | ● |
| Developed Framework | ● | ◐ | ◐ | ◐ | ● | ◐ | ◐ | ● |
| A: Activity-based K: Knowledge-based | | ●: Fully ◐: Partially | ○: Limited ○: Implicitly | * Only found in a few models | | | | |

is usually also incorporated by *Activity Networks* [45, 94], is currently not considered.

The organization has been slightly adapted compared to the baseline model, due to the addition of new activities. A one-to-one mapping (mirroring hypothesis) of design activities and responsibilities of engineers has been carried over from the baseline model. Responsibilities for the newly added validation activities, however, can be adjusted more freely. An organizational structure exists in the model, but it does not yet impact the efficiency of communication or knowledge sharing, which has been modeled by some *Activity-based Team Interaction* models [102, 104]. Competencies and expertise are captured for individual engineers, which impacts their efficiency and quality of work.

The use, creation, and transfer of knowledge is implemented in the model similar to the baseline model: Engineering activities require knowledge to be executed effectively and create new knowledge. This knowledge is then transferred through information exchange or collaboration.

The addition of tools significantly affects the simulation results, indicating a substantial

extension of the framework. Engineering tools can enable new activities and influence the overall outcome and performance of PD. This depends on the accuracy and interoperability of the tools, as well as the digital literacy of the engineers who use them. Currently, only the capabilities of virtual validation tools are considered in the model. These help identify quality issues earlier in the PD process. Other capabilities, such as optimization, generative design, engineering intelligence, or integrated design and analysis tools, are not included in the model. Therefore, engineering tools currently do not have a direct impact on the actual quality of designs, but only the accuracy of the measured quality.

Knowledge management and collaborative tools are modeled as centralized repositories that manage, store, and exchange knowledge and information. A limitation to this is that the model currently only supports one such tool, whereas, in most cases, multiple tools would be expected. Thus, the connection and interactions between different knowledge management tools cannot be analyzed.

9.2.2 Capabilities and Limitations

The validation of the hypotheses and the successful demonstration of the framework's capabilities indicate that the **Research Objective** with its defined modeling purpose (Section 3.3) has been achieved. This presents a significant contribution to the body of knowledge on product development and engineering management.

The simulation framework is a step towards more holistic PD simulations that can serve as decision support for planning and managing PD processes, as well as the implementation of digital tools. The intended usage of the framework, as defined in Section 3.5, is the evaluation of the later stages of a single PD project in the context of development and validation activities. The framework enables testing and evaluating different what-if scenarios or exploring the entire organizational design space. It also offers a high degree of flexibility to gain detailed insight into various aspects of the PD process. These can lead to the identification of inefficiencies or areas with the greatest potential for improvement.

However, it must be acknowledged that the model only simulates the configuration itself, not the cost and effort required to achieve changes. This means that the framework can currently only be used to explore the design space but not to conduct trade studies.

Another limitation of the framework is that several abstract parameters (e.g., accuracy, interoperability, digital literacy) have been introduced without providing any methods to quantify them. Lastly, no empirical validation was performed to confirm the practical applicability. Addressing these and other limitations could further enhance the framework in the future. The following section outlines several ways these limitations could be addressed to strengthen the framework.

9.3 Future Work

To address the limitations discovered and further develop the framework, several next steps can be taken. These are intended to increase performance, accuracy, credibility, and practical applicability. First, validation of the framework against real-world data is necessary. The functionality and usability can be improved by extending the scope, increasing the modeling flexibility, adding and integrating data sources, and extracting more relevant data. Ultimately, these efforts should lead to the development of a functional Digital Twin of an Organization (DTO).

9.3.1 Model Validation and Fine Tuning

Throughout the development of the framework, the intended behavior was continuously verified and the expected behavior was validated with respect to the hypotheses. However, this validation was based on a notional case study representing a simplified drone development project and observations made from the PD literature due to the lack of empirical data available. For higher accuracy and applicability of the framework, real-world data and experience must be used.

This could be approached through case studies on various development projects, begin-

ning with smaller, more accessible efforts (such as student projects) and gradually scaling up to complex large-scale projects. As projects grow in size, the value of running simulations is likely to increase because complex interactions and the emergent behavior become less predictable.

The validation of the sensitivity of different input parameters could also be done by consulting various domain experts and project managers. These could also help with further developing the framework, to be applicable in industry by providing the necessary modeling flexibility and the insights needed for planning projects in practice.

Abstract input and tuning parameters must also be quantifiable based on actual available data. Here, process mining and data analytics, or even a DTO, could help identify relevant factors and hidden dependencies. Methods for quantifying these parameters are essential to enable the cost-benefit analysis of potential changes to the organization or tools. This is required for the practical applicability of the framework.

A potential challenge to validating the framework is that ABS models are very difficult to validate and cannot easily prescribe a plan [73]. Some measures have been taken to mitigate these effects by extracting and visualizing a lot of the data generated by the model. For further validation and to generate more insights, these efforts must be continued.

9.3.2 Framework Improvements and Extensions

The framework could be extended in several ways, as highlighted in Table 9.1.

Allowing changes in process structures, such as overlapping and different sequences, would be beneficial, as different work policies can have a substantial effect on the lead time and cost of PD [45]. Enabling dynamic processes, previously proposed by [115], could change the process depending on the simulation context, leading to more realistic iteration and reduction in variability of simulation results. Reducing the variance is especially important to also increase the simulation accuracy.

Adding logic to the organizational domain could help explore the impact of different

organizational structures on knowledge sharing, communication, coordination, and decision making in PD. Less rigid mappings of the responsibilities for design activities would also improve modeling flexibility.

In the product domain, increasing the granularity by modeling individual activities for the requirements, functions, or interfaces of product elements could add more detail to the simulation by adding several activities per element (e.g., FEA and CFD for the propeller). Instead of an abstract quality value, the use of technical performance measures (TPMs) could also be beneficial [95, 141]. Adding system-level dependencies, similar to interfaces of components, would allow more detailed simulation of interactions during system design.

Digital tools could be modeled more comprehensively, by adding properties related to capabilities other than virtual validation, adding multiple knowledge management (EKM) tools and their interactions, or considering resource constraints caused by tool availability and licenses. Modeling learning for digital tools, similar to expertise and product knowledge, could also provide deeper insights into the potential risks of implementing new tools.

9.3.3 Development of a Digital Twin of an Organization

The vision of a Digital Twin of an Organization (DTO) is the broader context of this thesis. A DTO consists not only of a simulation model, but also requires a connection to data sources within the organization and should help optimize the organization [82, 84]. These extensions could enable the framework to be used for accurate cost-benefit analyzes instead of only evaluating a configuration without accounting for implementation costs.

Data Integration and Acquisition

For the simulation model function in the context of a DTO, data from the organization must be provided. Therefore, identifying relevant data sources is important. Possible examples of these data sources could be Enterprise Resource Planning (ERP) or Product Lifecycle Management (PLM) tools. Methods for extracting relevant process data through process

mining during PD have already been proposed [142, 143].

These approaches could generate large amounts of data, making the use of data analytics necessary. Insights generated from the data can then be used to contentiously optimize the organization and its PD process, but also refine and fine-tune the simulation model.

Furthermore, data related to the as-is and to-be architecture is required to simulate different configurations. In a preliminary version of the simulation framework, a methodology was proposed that uses an enterprise architecture and SysML to define input data for different configurations [134]. This approach could be further investigated to move towards a DTO of product development organizations when integrated with data.

Optimization of Product Development

A DTO would be used to optimize PD by continuously monitoring the process and providing recommendations to improve it [82]. The current framework is only able to evaluate a configuration and, therefore, is limited to manual iterative optimization.

In the future, an optimization algorithm for the simulation framework of this thesis could be developed. Optimizations have previously been developed for PD simulation models with smaller scopes [26] (e.g., [144–146]).

Since a changes to an organization (e.g., recruitment, training, tool implementation) cannot be immediately implemented and can have various external and internal constraints (e.g., cost, resources, cultural barriers), an optimization algorithm in the context of a DTO would need to account for these effects. The costs and barriers to implementing changes to an organization are not yet integrated into the simulation framework.

To make the simulation more viable on larger scales, especially for optimization, the framework itself will first have to be optimized by improving the coded implementation. The Monte Carlo simulation of the simple case study already required 15–25 minutes per configuration ($n_{runs} = 400$), indicating the need for improvement.

Appendices

APPENDIX A

BASELINE MODEL

Zhang and Thomson's model [106] was selected as the baseline model. This chapter serves as a short summary of the logic of the baseline model. In this thesis, several changes were made to the baseline model that are explained throughout Chapter 6 and Chapter 7.

Zhang and Thomson's model focuses on modeling the development of complex products from a knowledge perspective. It employs an agent-based modeling approach to simulate the learning and application of knowledge throughout the PD process. The model represents PD as a network of interdependent agents for product functions, design activities, and designers. These elements are connected through knowledge. The product is represented as a hierarchy of interdependent functions.

By simulating the interactions and activities at a micro-level, the model generates macro-level project performance measures, such as effort and duration.

A.1 Knowledge

In the model, knowledge is categorized into general knowledge and product knowledge. *General knowledge* refers to the technical expertise of a designer acquired through formal education and training. It determines the designer's ability to perform specific tasks and is classified into different levels of proficiency. *Product knowledge*, on the other hand, refers to a designer's understanding of the specific product being developed, which is accumulated through experience with similar projects. A higher level of product knowledge enables designers to anticipate potential integration issues and take proactive measures. The model also incorporates a learning curve effect, where designers gradually improve their efficiency, reducing the rework time as they gain experience.

A.2 Development Process

The overall process begins with the decomposition of the product into a hierarchy of functions and subsystems. Each function requires specific knowledge and effort to develop. Assigned designers develop the function through technical work. As functions are completed, they provide input to higher-level subsystems, which gradually lead to the full system design. Throughout the process, designers communicate and consult with each other to resolve uncertainties and ensure alignment. The model captures iterative cycles of design, review, and rework until the final product meets the required quality. The simulation follows a probabilistic approach, incorporating triangle distributions for task durations, communication delays, and rework probabilities, reflecting real-world uncertainties.

A.2.1 Technical Work

Technical work involves designers completing tasks based on their knowledge abilities and the complexity of the product functions. Each function is represented as a series of sequential tasks that must be completed before moving to higher levels of integration. The technical complexity of a function is determined by the types and severity of knowledge required. The designers' work efficiency is influenced by their competence, defined by their general knowledge with respect to the knowledge requirements of the function. The quality of the work performed depends on the competence and product knowledge of the designer. Completed tasks undergo a technical review to check quality, triggering rework if necessary. Integration of higher-level functions begins only when all associated subsystems meet a predefined quality threshold.

A.2.2 Communication and Consultation

Communication plays a crucial role in ensuring the alignment among interdependent functions. It involves the preparation, exchange, and reading of design information, such as

task outcomes and interface data, between dependent functions.

Consultation is modeled as meetings between designers to address technical difficulties and interface issues. These meetings are triggered when problems occur during technical work or interface incompatibilities are detected during communication. During consultation, designers' knowledge is increased, leading to higher quality work.

APPENDIX B
MODEL PARAMETERS

B.1 Knowledge Domains and Interface Types

Table B.1: Interface types and possible severity values (based on [36]).

| Interface Type | No Interaction | Weak | Strong | Severe |
|-----------------------|----------------|------|--------|--------|
| Spatial/Geometric (G) | 0 | 1 | 3 | 9 |
| Structural (S) | 0 | 1 | 3 | 9 |
| Energy (E) | 0 | 1 | 3 | 9 |
| Material (M) | 0 | 1 | 3 | 9 |
| Data/Information (D) | 0 | 1 | 3 | 9 |

Table B.2: Knowledge domains and their possible values used for defining knowledge requirements of product elements or digital tools, and the expertise of engineers.

| Knowledge Domains | None | Basic | Advanced | Expert |
|--|------|-------|----------|--------|
| k_1 or a_1 Mechanical Engineering | 0 | 1 | 2 | 3 |
| k_2 or a_2 Aerospace Engineering | 0 | 1 | 2 | 3 |
| k_3 or a_3 Aerodynamics | 0 | 1 | 2 | 3 |
| k_4 or a_4 Electrical Engineering | 0 | 1 | 2 | 3 |
| k_5 or a_5 Software Engineering | 0 | 1 | 2 | 3 |
| k_6 or a_6 Stability and Control | 0 | 1 | 2 | 3 |
| k_7 or a_7 Sensor Technology | 0 | 1 | 2 | 3 |
| k_8 or a_8 Signal Processing | 0 | 1 | 2 | 3 |
| k_9 or a_9 Embedded Systems | 0 | 1 | 2 | 3 |
| k_{10} or a_{10} Manufacturing | 0 | 1 | 2 | 3 |
| DL_{Eng} Engineering Tool Digital Literacy | 0 | 1 | 2 | 3 |
| DL_{EKM} EKM Tool Digital Literacy | 0 | 1 | 2 | 3 |

B.2 Tuning Parameters and Settings

Table B.3: Tuning parameters and settings of the simulation model.

| Parameter | | Value |
|--|--|------------|
| – | Excess efficiency (> 1) factor, applied to Equation 7.9 and 7.10 | 0.2 |
| <i>Product Definition</i> | | |
| r | Upper limit of the knowledge scale | 3 |
| IS_{max} | Upper limit of for interface severity | 9 |
| <i>Quality</i> | | |
| α_g | Shape parameter for the solution goodness (Equation 7.13) | 50 |
| β_g | Shape parameter for the solution goodness (Equation 7.13) | 10 |
| <i>Collaboration and Consultation</i> | | |
| η_{collab} | Collaboration efficiency | 1.2 |
| E_{collab} | Collaboration effort (min, max) | 2–3 h |
| η_{cnslt} | Consultation efficiency | 0.5 |
| E_{cnslt} | Consultation effort (min, max) | 0.5–1 h |
| <i>EKM tool and information exchange</i> | | |
| η_{EKM} | Knowledge base efficiency | 0.3 |
| E_{EKM} | Knowledge base latency (min, max) | 0.5–1 h |
| e_{info} | Information handling effort per unit of information (min, max) | 0.2–0.3 h |
| e_{verif} | Verification effort per unit of information (min, max) | 0.3–0.45 h |
| $P_{I,b}$ | Base information exchange probability | 0.1 |
| $P_{I,f}$ | Information exchange probability factor | 0.5 |
| <i>Requirements and Risk Parameters</i> | | |
| T_Q | Quality Target | 0.9 |
| T_{LT} | Lead Time Target | 2 years |
| T_C | Cost Target | \$1.4 m |
| κ_{LT} | Lead Time Risk factor (Equation 3.1) | \$20k/wk |
| κ_C | Cost Risk factor (Equation 3.2) | \$10/\$k |
| <i>Simulation Settings</i> | | |
| E_{td} | Task duration | 4 h |
| Δt | Step size | 0.1 h |
| n_{runs} | Number of runs (Monte Carlo) | 400 |
| – | Work hours per week | 40 h |

APPENDIX C

INPUT DATA OF THE NOTIONAL CASE STUDY

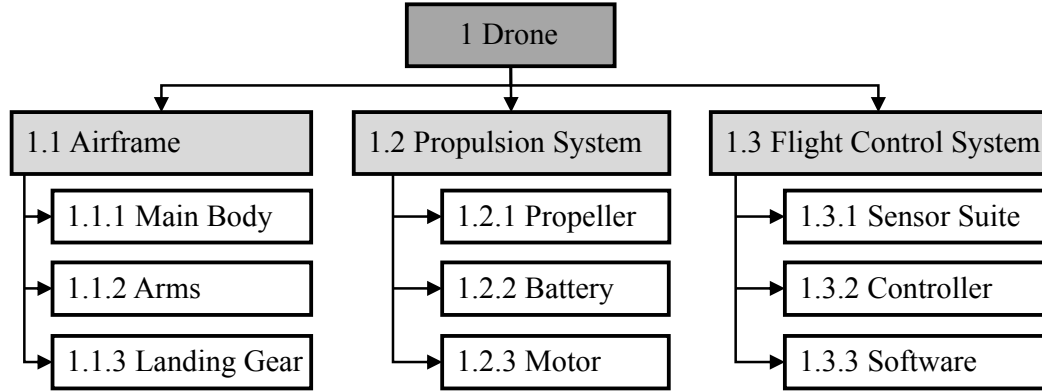


Figure C.1: Hierarchical product architecture of the notional drone.

Table C.1: Properties of the defined product elements.

| Product Element | Knowledge Requirement \vec{K}_R | | | | | | | | | | Novelty | Imp. | Procure |
|--------------------|-----------------------------------|-------|-------|-------|-------|-------|-------|-------|-------|----------|---------|------|---------|
| | k_1 | k_2 | k_3 | k_4 | k_5 | k_6 | k_7 | k_8 | k_9 | k_{10} | | | |
| 1 Drone | 1 | 3 | 2 | 1 | 1 | 2 | 0 | 0 | 0 | 1 | 0.5 | 1 | |
| 1.1 Airframe | 3 | 1 | 2 | 0 | 0 | 1 | 0 | 0 | 0 | 2 | 0.5 | 0.3 | |
| 1.1.1 Main Body | 2 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0.5 | 0.15 | |
| 1.1.2 Arms | 3 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0.5 | 0.1 | |
| 1.1.3 Landing Gear | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 0.5 | 0.05 | |
| 1.2 Propulsion | 2 | 3 | 2 | 2 | 0 | 1 | 0 | 0 | 0 | 1 | 0.5 | 0.3 | |
| 1.2.1 Propeller | 3 | 2 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0.5 | 0.1 | |
| 1.2.2 Battery | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 1 | 0 | 0.5 | 0.1 | Yes |
| 1.2.3 Motor | 2 | 0 | 0 | 2 | 1 | 0 | 1 | 0 | 0 | 0 | 0.5 | 0.1 | Yes |
| 1.3 Flight Control | 0 | 2 | 2 | 1 | 2 | 3 | 1 | 2 | 1 | 0 | 0.5 | 0.4 | |
| 1.3.1 Sensor Suite | 0 | 0 | 0 | 2 | 1 | 1 | 3 | 2 | 1 | 0 | 0.5 | 0.1 | Yes |
| 1.3.2 Controller | 0 | 0 | 0 | 3 | 1 | 0 | 1 | 2 | 3 | 0 | 0.5 | 0.05 | |
| 1.3.3 Software | 0 | 2 | 1 | 1 | 3 | 3 | 1 | 2 | 1 | 0 | 0.5 | 0.25 | |

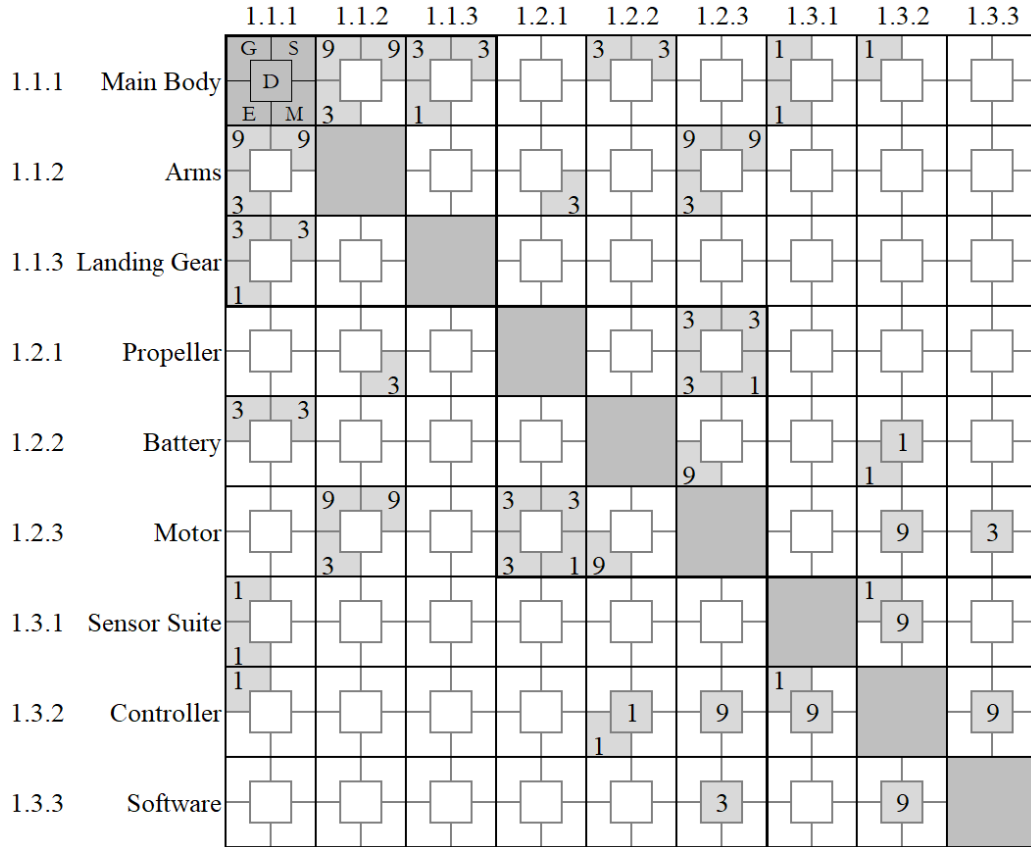


Figure C.2: Design structure matrix of the defined drone showing the interfaces between components and their severity.

Table C.2: Duration and learning rates for the high-level PD activities.

| Activity | Duration ^a | | | Learning rate l |
|-------------------------|-----------------------|------|-----|-------------------|
| | Min | Mode | Max | |
| System Design | 10 | 15 | 20 | 0.8 |
| LF System Simulation | 10 | 12 | 14 | 0.9 |
| Design | 20 | 25 | 30 | 0.8 |
| Component Simulation | 10 | 14 | 18 | 0.9 |
| Virtual Integration | 4 | 6 | 8 | 0.6 |
| HF System Simulation | 6 | 8 | 10 | 0.9 |
| Prototype Manufacturing | 36 | 40 | 44 | 0.95 |
| Component Testing | 12 | 15 | 18 | 0.9 |
| Physical Integration | 18 | 20 | 22 | 0.95 |
| System Testing | 24 | 30 | 36 | 0.9 |

^a h/complexity

Table C.3: Responsibilities assigned to the engineer agents.

| Tool | Responsibilities |
|-------------------------------------|---|
| System Team | |
| Systems Engineer 1 | System Design & Virtual Integration (Drone) |
| Systems Engineer 2 | System Design & Virtual Integration (Propulsion System) |
| Systems Engineer 3 | System Design & Virtual Integration (Flight Control System) |
| Simulation Engineer 1 | LF System Simulation (all), Component Simulation (Controller, Control Software) |
| Simulation Engineer 2 | HF System Simulation (all) |
| Design Team | |
| Design Engineer 1 | System Design & Virtual Integration (Airframe) |
| Design Engineer 2 | Design (Main Body) |
| Design Engineer 3 | Design (Landing Gear) |
| Design Engineer 4 | Design (Arms) |
| Design Engineer 5 | Design (Propeller) |
| Structural Engineer 1 | Component Simulation (Main Body), HF System Simulation (Airframe) |
| Structural Engineer 2 | Component Simulation (Arms, Landing Gear, Propeller) |
| Electronics Team | |
| Electrical Engineer 1 | Design (Motor) |
| Electrical Engineer 2 | Design (Sensor Suite) |
| Electrical Engineer 3 | Design (Controller) |
| Electrical Engineer 4 | Design (Battery) |
| Software Engineer 1 | Design & Prototyping & Testing (Control Software) |
| Validation Team | |
| Prototyping Engineer 1 | Prototyping (Components) |
| Prototyping Engineer 2 | Prototyping (Subsystems) |
| Prototyping Engineer 3 | Prototyping (Drone) |
| Testing Engineer 1 | Testing (Components) |
| Testing Engineer 2 | Testing (Subsystems, Drone) |
| Supplier Proxies^a | |
| <i>Supplier 1, 2, 3</i> | Prototyping (Motor, Battery, Sensor Suite) |

^a Needed to generate activities related to procurement.

Table C.4: Baseline properties of the defined engineer agents.

| Engineer Agent | Expertise | | | | | | | | | | Digital Literacy | | Experience | Salary (\$k) |
|-----------------------|-----------|-------|-------|-------|-------|-------|-------|-------|-------|----------|------------------|------------|------------|--------------|
| | a_1 | a_2 | a_3 | a_4 | a_5 | a_6 | a_7 | a_8 | a_9 | a_{10} | DL_{Eng} | DL_{EKM} | | |
| System Team | | | | | | | | | | | | | | |
| Systems Engineer 1 | 0.5 | 2 | 1 | 1 | 2 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 0.5 | 150 |
| Systems Engineer 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 1 | 2 | 2 | 0.5 | 140 |
| Systems Engineer 3 | 0.5 | 3 | 1 | 1 | 2 | 3 | 1 | 1 | 1 | 0.5 | 2 | 2 | 0.5 | 140 |
| Simulation Engineer 1 | 2 | 2 | 1 | 1 | 2 | 2 | 1 | 1 | 1 | 0.5 | 2 | 2 | 0.5 | 140 |
| Simulation Engineer 2 | 2 | 3 | 2 | 2 | 2 | 3 | 1 | 2 | 1 | 0.5 | 2 | 2 | 0.5 | 140 |
| Design Team | | | | | | | | | | | | | | |
| Design Engineer 1 | 3 | 1 | 1 | 0.5 | 1 | 2 | 0.5 | 0.5 | 0.5 | 2 | 2 | 2 | 0.5 | 150 |
| Design Engineer 2 | 3 | 1 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 3 | 2 | 2 | 0.5 | 140 |
| Design Engineer 3 | 3 | 1 | 2 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 2 | 2 | 2 | 0.5 | 140 |
| Design Engineer 4 | 3 | 1 | 2 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 2 | 2 | 2 | 0.5 | 140 |
| Design Engineer 5 | 2 | 1 | 2 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 2 | 2 | 2 | 0.5 | 140 |
| Structural Engineer 1 | 3 | 2 | 3 | 0.5 | 1 | 2 | 1 | 1 | 0.5 | 2 | 2 | 2 | 0.5 | 140 |
| Structural Engineer 2 | 3 | 2 | 2 | 0.5 | 1 | 0.5 | 1 | 1 | 0.5 | 2 | 2 | 2 | 0.5 | 140 |

Continued on next page

Table C.4: Baseline properties of the defined engineer agents. (Continued)

| Engineer Agent | Expertise | | | | | | | | | | Digital Literacy | | Experience | Salary (\$k) |
|-------------------------|-----------|-------|-------|-------|-------|-------|-------|-------|-------|----------|------------------|------------|------------|--------------|
| | a_1 | a_2 | a_3 | a_4 | a_5 | a_6 | a_7 | a_8 | a_9 | a_{10} | DL_{Eng} | DL_{EKM} | | |
| Electronics Team | | | | | | | | | | | | | | |
| Electrical Engineer 1 | 0.5 | 2 | 0.5 | 3 | 2 | 2 | 2 | 2 | 1 | 0.5 | 2 | 2 | 0.5 | 150 |
| Electrical Engineer 2 | 0.5 | 0.5 | 0.5 | 2 | 1 | 2 | 3 | 1 | 2 | 0.5 | 2 | 2 | 0.5 | 140 |
| Electrical Engineer 3 | 0.5 | 0.5 | 0.5 | 2 | 1 | 2 | 2 | 1 | 3 | 0.5 | 2 | 2 | 0.5 | 140 |
| Electrical Engineer 4 | 0.5 | 0.5 | 0.5 | 3 | 1 | 2 | 1 | 1 | 2 | 0.5 | 2 | 2 | 0.5 | 140 |
| Software Engineer 1 | 0.5 | 2 | 0.5 | 1 | 3 | 2 | 1 | 1 | 1 | 0.5 | 2 | 2 | 0.5 | 160 |
| Validation Team | | | | | | | | | | | | | | |
| Prototyping Engineer 1 | 3 | 2 | 0.5 | 1 | 1 | 1 | 0.5 | 2 | 2 | 3 | 2 | 2 | 0.5 | 90 |
| Prototyping Engineer 2 | 2 | 2 | 1 | 2 | 1 | 1 | 2 | 1 | 2 | 2 | 2 | 2 | 0.5 | 90 |
| Prototyping Engineer 3 | 1 | 2 | 0.5 | 2 | 1 | 1 | 2 | 1 | 2 | 3 | 2 | 2 | 0.5 | 90 |
| Testing Engineer 1 | 2 | 3 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 2 | 2 | 0.5 | 90 |
| Testing Engineer 2 | 2 | 1 | 0.5 | 2 | 1 | 1 | 1 | 2 | 2 | 1 | 2 | 2 | 0.5 | 90 |

Table C.5: All defined tools and their capabilities.

| Tool | Capabilities |
|--------------------------------|--|
| Baseline Tools | |
| MBSE | System Design (Drone, Airframe, Propulsion System, Flight Control System) |
| Basic System Simulation | LF System Simulation (Drone, Propulsion System, Flight Control System) |
| MCAD | Design (Main Body, Arms, Landing Gear, Propeller), Virtual Integration (Airframe) |
| ECAD | Design (Controller) |
| IDE | All Activities (Control Software) |
| FEA | All Simulation Activities (Airframe, Main Body, Arms, Landing Gear, Propeller) |
| Circuit Simulation | Component Simulation (Controller) |
| Manufacturing Equipment | Prototyping (Main Body, Arms, Landing Gear, Propeller, Controller) |
| Assembly Equipment | Prototyping (Drone, Airframe, Propulsion System, Flight Control System) |
| Universal Test Machine | Testing (Airframe, Main Body, Arms, Landing Gear, Propeller) |
| HIL Equipment | Testing (Flight Control System, Controller) |
| System Testing Facility | Testing (Drone, Propulsion System) |
| New Tool | |
| Advanced System Simulation | Virtual Integration and HF System Simulation (Drone, Propulsion System, Flight Control System) |
| Proxy Tools^a | |
| <i>Supply Chain Management</i> | Design (Battery, Motor, Sensor Suite) |
| <i>Supplier</i> | Prototyping (Battery, Motor, Sensor Suite) |

^a Needed to generate activities related to procurement.

Table C.6: Baseline properties of the defined tools.

| Tool | Cost (\$k) | Productivity T_p | Usability T_{Usab} | Accuracy T_{Acc} | Interoperability T_I |
|--------------------------------|------------|--------------------|----------------------|--------------------|------------------------|
| Baseline Tools | | | | | |
| MBSE | 10.0 /mo | 0.7 | 2 | 0.1 ^a | 0.5 |
| Basic System Simulation | 5.0 /mo | 0.7 | 2 | 0.4 | 0.5 |
| MCAD | 4.0 /mo | 0.7 | 2 | 0.1 | 0.5 |
| ECAD | 4.0 /mo | 0.7 | 2 | 0.1 | 0.5 |
| IDE | 2.0 /mo | 0.7 | 2 | 0.6 | 0.5 |
| FEA | 12.0 /mo | 0.7 | 2 | 0.6 | 0.5 |
| Circuit Simulation | 4.0 /mo | 0.7 | 2 | 0.6 | 0.5 |
| Manufacturing Equipment | 0.2 /h | 0.7 | – | – | 0.5 |
| Assembly Equipment | 0.1 /h | 0.7 | – | – | 0.5 |
| Universal Test Machine | 0.1 /h | 0.7 | – | 0.95 | 0.5 |
| HIL Equipment | 0.1 /h | 0.7 | – | 0.95 | 0.5 |
| System Testing Facility | 0.2 /h | 0.7 | – | 0.95 | 0.5 |
| New Tool | | | | | |
| Advanced System Simulation | 15.0 /mo | 0.7 | 2 | 0.1...0.9 | 0.5 |
| Proxy Tools^b | | | | | |
| <i>Supply Chain Management</i> | 1.0 /mo | 1.0 | 2 | – | 0.5 |
| <i>Supplier</i> | 0.3 /h | 1.0 | – | – | 0.5 |
| EKM Tool | 5.0 /mo | 0.7 | 2 | – | – |

^a The accuracy of the MBSE tool relates to the fidelity, completeness, and traceability of models created by the tool.

^b Needed to generate activities related to procurement.

APPENDIX D

SIMULATION CODE AND DATA

The simulation model was fully implemented using Python version 3.12. The logic, functionalities, and most important parameters were discussed throughout the thesis. The code, as well as the data generated by all the experiments of this thesis, can be found on GitHub:

`https://github.com/SeanRM2000/Product-Development-Simulation`

The code is divided into several scripts. The main simulation logic is implemented in «*sim_run.py*», and the Monte Carlo simulation logic is implemented in «*monte_carlo.py*». The simulation inputs are stored in several json and csv files. Each configuration is stored in a separate folder inside the *Architecture/Inputs* folder. The simulation results are stored in the *Architecture/Outputs* folder. A detailed summary of the simulation results can be found in «*DOE_Results.xlsx*» in the folder *DOE*.

The simulation settings and tuning parameters are defined in «*sim_settings.py*» and «*tuning_params.py*». The remaining scripts are used for preprocessing of the input data to create all agents, as well as the detailed PD process.

REFERENCES

- [1] T. Tomiyama, E. Lutters, R. Stark, and M. Abramovici, “Development capabilities for smart products,” *CIRP Annals*, vol. 68, no. 2, pp. 727–750, 2019.
- [2] T. J. Marion and S. K. Fixson, “The transformation of the innovation process: How digital tools are changing work, collaboration, and organizations in new product development,” *Journal of Product Innovation Management*, vol. 38, no. 1, pp. 162–215, 2021.
- [3] R. Jaifer, Y. Beauregard, and N. Bhuiyan, “New framework for effort and time drivers in aerospace product development projects,” *Engineering Management Journal*, vol. 33, no. 2, pp. 76–95, 2021.
- [4] S. D. Eppinger and A. R. Chitkara, “The new practice of global product development,” *MIT Sloan Management Review*, vol. 47, no. 4, pp. 22–30, 2006.
- [5] J. P. Stefanovitz and A. B. L. de Sousa Jabbour, “Product development management complexity: Emerging challenges and the role of senior leadership,” *Journal of Knowledge Management*, vol. 26, no. 7, pp. 1676–1686, 2022.
- [6] M. Jankovic and A. M. Hein, “Architecting engineering systems: Designing critical interfaces,” in *Handbook of Engineering Systems Design*, A. Maier, J. Oehmen, and P. E. Vermaas, Eds., Cham: Springer, 2023, pp. 1–25.
- [7] Q. Yang, S. Kherbachi, Y. S. Hong, and C. Shan, “Identifying and managing coordination complexity in global product development project,” *International Journal of Project Management*, vol. 33, no. 7, pp. 1464–1475, 2015.
- [8] J. M. Costa, H. Rozenfeld, C. S. T. Amaral, R. M. Marcacinit, and S. O. Rezende, “Systematization of recurrent new product development management problems,” *Engineering Management Journal*, vol. 25, no. 1, pp. 19–34, 2013.
- [9] R. G. Cooper, “The drivers of success in new-product development,” *Industrial Marketing Management*, vol. 76, pp. 36–47, 2019.
- [10] H. Liu, Y. Luo, J. Geng, and P. Yao, “Research hotspots and frontiers of product R&D management under the background of the digital intelligence era — bibliometrics based on citespace and histcite,” *Applied Sciences*, vol. 11, no. 15, 2021.
- [11] I. F. Arromba *et al.*, “Industry 4.0 in the product development process: Benefits, difficulties and its impact in marketing strategies and operations,” *Journal of Business & Industrial Marketing*, vol. 36, no. 3, pp. 522–534, 2021.

- [12] T. Mauerhoefer, S. Strese, and M. Brettel, “The impact of information technology on new product development performance,” *Journal of Product Innovation Management*, vol. 34, no. 6, pp. 719–738, 2017.
- [13] K. Henderson, T. McDermott, E. V. Aken, and A. Salado, “Towards developing metrics to evaluate digital engineering,” *Systems Engineering*, vol. 26, pp. 3–31, 2023.
- [14] B. Clayton *et al.*, “The impact of digital engineering on the defense acquisition and the supply chain: Insights form an industry survey,” RAND Corporation, Tech. Rep., 2024.
- [15] Department of Defense, "Digital engineering strategy," 2018, [url:https://man.fas.org/eprint/digeng-2018.pdf](https://man.fas.org/eprint/digeng-2018.pdf) .
- [16] F. Tao, X. Ma, W. Liu, and C. Zhang, “Digital engineering: State-of-the-art and perspectives,” *Digital Engineering*, vol. 1, 2024.
- [17] H. Stoewer, “Perspectives on SE, MBSE, and digital engineering: Road to a digital enterprise,” in *Handbook of Model-Based Systems Engineering*, A. M. Madni, N. Augustine, and M. Sievers, Eds., Cham: Springer, 2023, pp. 1–37.
- [18] M. V. Pereira Pessôa and J. M. Jauregui Becker, “Smart design engineering: A literature review of the impact of the 4th industrial revolution on product design and development,” *Research in Engineering Design*, vol. 31, pp. 175–195, 2020.
- [19] L. Roucoules and N. Anwer, “Coevolution of digitalisation, organisations and product development cycle,” *CIRP Annals — Manufacturing Technology*, vol. 70, pp. 519–542, 2021.
- [20] P. Mugge, H. Abbu, T. L. Michaelis, A. Kwiatkowski, and G. Gudergan, “Patterns of digitization: A practical guide to digital transformation,” *Research-Technology Management*, vol. 63, no. 2, pp. 27–35, 2020.
- [21] W. Majerowicz and S. A. Shinn, “Schedule matters: Understanding the relationship between schedule delays and costs on overruns,” *IEEE Aerospace Conference*, pp. 1–8, 2016,
- [22] B. Gozluklu and J. Serman, “System dynamics to understand and improve the performance of complex projects,” in *Research Handbook on Complex Project Organizing*, G. M. Winch, M. Brunet, and D. Cao, Eds., Cheltenham: Edward Elgar, 2023, pp. 70–77.

- [23] J. Meluso, J. Austin-Breneman, J. P. Bagrow, and L. Hébert-Dufresne, “A review and framework for modeling complex engineered system development processes,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 12, pp. 7679–7691, 2022.
- [24] N. P. Whitehead, T. Light, A. Luna, and J. Mignano, “A framework for assessing the costs and benefits of digital engineering: A systems approach,” RAND Corporation, Tech. Rep., 2024.
- [25] S. Schweigert-Recksiek, H. Riener, P. Koch, D. Meier, A. Daners, and M. Krastel, “Handling the complexity of tool selection processes – simulation data management in the automotive supplier industry,” *Proceedings of the 17th International DSM Conference*, pp. 88–97, 2022.
- [26] A. A. Yassine, “Managing the development of complex product systems: An integrative literature review,” *IEEE Transactions on Engineering Management*, vol. 68, no. 6, pp. 1619–1636, 2021.
- [27] Q. Yang and P. Tian, “New product development project management: Insights and research tendency from a bibliometric-based literature review,” *Concurrent Engineering: Research and Applications*, vol. 31, no. 1-2, pp. 65–79, 2023.
- [28] "Design of technical products and systems: Model of product design," VDI 2221, Verein Deutscher Ingenieure, Düsseldorf, DE, Nov., 2019.
- [29] K. T. Ulrich and S. D. Eppinger, *Product Design and Development*, 5th ed. New York: McGraw-Hill, 2012.
- [30] H. Negele, E. Fricke, and E. Igenbergs, “ZOPH – a systemic approach to the modeling of product development systems,” *System Engineering*, vol. 7, no. 1, pp. 266–273, 1997.
- [31] T. R. Browning, E. Fricke, and H. Negele, “Key concepts in modeling product development processes,” *System Engineering*, vol. 9, no. 2, pp. 104–128, 2006.
- [32] D. M. Sharman and A. A. Yassine, “Characterizing complex product architectures,” *Systems Engineering*, vol. 7, no. 1, pp. 35–60, 2004.
- [33] T. R. Browning, “Applying the design structure matrix to system decomposition and integration problems: A review and new directions,” *IEEE Transactions on Engineering Management*, vol. 48, no. 3, pp. 292–306, 2001.
- [34] S. Jung, O. Asikoglu, and T. W. Simpson, “A method to evaluate direct and indirect design dependencies between components in a product architecture,” *Research in Engineering Design*, vol. 29, pp. 507–530, 2018.

- [35] T. U. Pimmler and S. D. Eppinger, “Integration analysis of product decompositions,” *Proceedings of the ASME 1994 Design Technical Conferences collocated with the ASME 1994 International Computers in Engineering Conference and Exhibition and the ASME 1994 8th Annual Database Symposium. 6th International Conference on Design Theory and Methodology*, pp. 343–351, 1994.
- [36] S. D. Eppinger, N. R. Joglekar, A. Olechowski, and T. Teo, “Improving the systems engineering process with multilevel analysis of interactions,” *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, vol. 28, pp. 323–337, 2014.
- [37] X. Zhang and V. Thomson, “A knowledge-based measure of product complexity,” *Computers & Industrial Engineering*, vol. 115, pp. 80–87, 2018.
- [38] R. G. Cooper, “Stage-gate systems: A new tool for managing new products,” *Business Horizons*, vol. 33, no. 3, pp. 44–54, 1990.
- [39] R. G. Cooper, “What’s next?: After stage gate,” *Research-Technology Management*, vol. 57, no. 1, pp. 20–31, 2014.
- [40] D. Unger and S. Eppinger, “Improving product development process design: A method for managing information flows, risks, and iterations,” *Journal of Engineering Design*, vol. 22, no. 10, pp. 689–699, 2011.
- [41] D. D. Walden *et al.*, *INCOSE Systems Engineering Handbook*. San Diego, USA: International Council on Systems Engineering, 2023, vol. 5.
- [42] S. Vajna, “Workflow for design,” in *Design process improvement: A review of current practice*, J. Clarkson and C. Eckert, Eds., London: Springer, 2005, pp. 366–385.
- [43] D. C. Wynn and C. M. Eckert, “Perspectives on iteration in design and development,” *Research in Engineering Design*, vol. 28, pp. 153–184, 2017.
- [44] C. Meier, T. R. Browning, A. A. Yassine, and U. Walter, “The cost of speed: Work policies for crashing and overlapping in product development projects,” *IEEE Transactions on Engineering Management*, vol. 62, no. 2, pp. 237–255, 2015.
- [45] J. F. Maier, D. C. Wynn, W. Biedermann, U. Lindemann, and P. J. Clarkson, “Simulating progressive iteration, rework and change propagation to prioritise design tasks,” *Research in Engineering Design*, vol. 25, pp. 283–307, 2014.
- [46] M. E. Sosa and J. Mihm, “Organization design for new product development,” in *Handbook of New Product Development Management*, C. H. Loch and S. Kavadias, Eds., New York, NY: Taylor & Francis, 2008, pp. 165–197.

- [47] M. E. Sosa, S. D. Eppinger, and C. M. Rowles, “The misalignment of product architecture and organizational structure in complex product development,” *Management Science*, vol. 50, no. 12, pp. 1674–1689, 2004.
- [48] A. Cabigiosu and A. Camuffo, “Beyond the “mirroring” hypothesis: Product modularity and interorganizational relations in the air conditioning industry,” *Organization Science*, vol. 23, no. 3, pp. 686–703, 2012.
- [49] L. J. Colfer and C. Y. Baldwin, “The mirroring hypothesis: Theory, evidence, and exceptions,” *Industrial and Corporate Change*, vol. 25, no. 5, pp. 709–738, 2016.
- [50] M. E. Conway, “How do committees invent?” *Datamation*, vol. 14, no. 4, pp. 28–31, 1968.
- [51] P. Parraguez, S. D. Eppinger, and A. M. Maier, “Information flow through stages of complex engineering design projects: A dynamic network analysis approach,” *IEEE Transactions on Engineering Management*, vol. 62, no. 4, pp. 300–314, 2015.
- [52] V. Hubka and W. E. Eder, *Design Science: Introduction to the Needs, Scope and Organization of Engineering Design Knowledge*. London: Springer, 1996.
- [53] D. C. Wynn and P. J. Clarkson, “Improving the engineering design process by simulating iteration impact with asm2.0,” *Research in Engineering Design*, vol. 32, pp. 127–156, 2021.
- [54] A. A. Yassine and J. B. Bradley, “A knowledge-driven, network-based computational framework for product development systems,” *Journal of Computing and Information Science in Engineering*, vol. 13, 2013.
- [55] M. Cantamessa and F. Montagna, *Management of Innovation and Product Development*. London: Springer, 2023.
- [56] I. Nonaka, “A dynamic theory of organizational knowledge creation,” *Organization Science*, vol. 5, no. 1, pp. 14–37, 2015.
- [57] E. A. Smith, “The role of tacit and explicit knowledge in the workplace,” *Journal of Knowledge Management*, vol. 5, no. 4, pp. 311–321, 2001.
- [58] P. Heisig, N. H. Caldwell, and P. J. Clarkson, “Core information categories for engineering design – contrasting empirical studies with a review of integrated models,” *Journal of Engineering Design*, vol. 25, no. 1-3, pp. 88–124, 2014.
- [59] S. K. Chandrasegaran *et al.*, “The evolution, challenges, and future of knowledge representation in product design systems,” *Computer-Aided Design*, vol. 45, pp. 204–228, 2013.

- [60] H. Qin, H. Wang, and A. Johnson, “Understanding the information needs and information-seeking behaviours of new-generation engineering designers for effective knowledge management,” *Journal of Information Management*, vol. 72, no. 6, pp. 853–868, 2020.
- [61] F. Deken, M. Kleinsmann, M. Aurisicchio, K. Lauche, and R. Bracewell, “Tapping into past design experiences: Knowledge sharing and creation during novice–expert design consultations,” *Research in Engineering Design*, vol. 23, pp. 203–218, 2012.
- [62] L. Argote and E. Fahrenkopf, “Knowledge transfer in organizations: The roles of members, tasks, tools, and networks,” *Organizational Behavior and Human Decision Processes*, vol. 136, pp. 146–159, 2016.
- [63] N. Chucholowski and U. Lindemann, “An initial metamodel to evaluate potentials for graph-based analyses of product development projects,” *Proceedings of the 17th International DSM Conference*, pp. 75–88, 2015.
- [64] R. Stark, *Virtual Product Creation in Industry, The Difficult Transformation from IT Enabler Technology to Core Engineering Competence*. Berlin: Springer, 2022.
- [65] S. Thomke and T. Fujimoto, “The effect of “front-loading” problem-solving on product development performance,” *Journal of Product Innovation Management*, vol. 17, pp. 128–142, 2000.
- [66] N. Hutchison *et al.*, “Developing the digital engineering competency framework (DECF) – Phase 2,” Stevens Institute of Technology, Tech. Rep. SERC-2021-TR-005, 2021.
- [67] "Digital engineering and modeling practice", MIL-HDBK-539, Department of Defense, Washington, DC, USA, 2022.
- [68] AIAA Digital Engineering Integration Committee, “Digital thread: Definition, value, and reference model,” AIAA, AIA, and NAFEMS, Tech. Rep., 2023.
- [69] A. L. Szejka, O. C. Jr., H. Panetto, E. R. Loures, and A. Aubry, “Semantic interoperability for an integrated product development process: A systematic literature review,” *International Journal of Production Research*, vol. 55, no. 22, pp. 6691–6709, 2017.
- [70] C. M. Eckert and P. J. Clarkson, “Planning development processes for complex products,” *Research in Engineering Design*, vol. 21, pp. 153–171, 2010.
- [71] R. Pellerin and N. Perrier, “A review of methods, techniques and tools for project planning and control,” *International Journal of Production Research*, vol. 57, no. 7, pp. 2160–2178, 2019.

- [72] D. C. Wynn and P. J. Clarkson, "Process models in design and development," *Research in Engineering Design*, vol. 29, pp. 161–202, 2018.
- [73] D. C. Wynn and P. J. Clarkson, *The Design and Development Process: Perspectives, Approaches and Models*. Cham: Springer Nature, 2024.
- [74] J. Bankauskaite, "Comparative analysis of enterprise architecture frameworks," *CEUR workshop proceedings: IVUS 2019 international conference on information technologies: proceedings of the international conference on information technologies*, vol. 2470, pp. 61–64, 2019.
- [75] U. Lindemann and M. Maurer, "Facing multi-domain complexity in product development," *The Future of Product Development*, pp. 351–361, 2007.
- [76] M. Danilovic and T. R. Browning, "Managing complex product development projects with design structure matrices and domain mapping matrices," *International Journal of Project Management*, vol. 25, pp. 300–314, 2007.
- [77] B. O'Donovan, C. Eckert, J. Clarkson, and T. R. Browning, "Design planning and modelling," in *Design process improvement: A review of current practice*, J. Clarkson and C. Eckert, Eds., London: Springer, 2005, pp. 60–87.
- [78] P. Ballesteros-Pérez, G. D. Larsen, and M. C. González-Cruz, "Do projects really end late? On the shortcomings of the classical scheduling techniques," *Journal of Technology and Science Education*, vol. 8, no. 1, pp. 17–33, 2018.
- [79] T. R. Browning, "Building models of product development processes: An integrative approach to managing organizational knowledge," *System Engineering*, vol. 21, no. 1, pp. 70–87, 2018.
- [80] R. K. Jonkers and K. E. Shahroudi, "A design change, knowledge, and project management flight simulator for product and project success," *IEEE Systems Journal*, vol. 15, no. 1, pp. 1130–1139, 2021.
- [81] R. Parmar, A. Leiponen, and L. D. Thomas, "Building an organizational digital twin," *Business Horizons*, vol. 63, no. 6, pp. 725–736, 2020.
- [82] M. C. Becker and B. T. Pentland, "Digital twin of an organization: Are you serious?" In *Business Process Management Workshops*, A. Marrella and B. Weber, Eds., Cham: Springer, 2022, pp. 243–254.
- [83] K. Lyytinen, B. Weber, M. C. Becker, and B. T. Pentland, "Digital twins of organization: Implications for organization design," *Journal of Organization Design*, 2023.

- [84] W. M. P. van der Aalst, “Concurrency and objects matterdisentangling the fabric of real operational processes to create digital twins,” *Theoretical Aspects of Computing – ICTAC 2021*, pp. 3–17, 2021.
- [85] A. Aamer, A. Zadeh, P. Mali, and C. Bolick, “Emerging technologies and principle-based project management: A systematic literature review and research agenda,” *Management Review Quarterly*, 2024.
- [86] B. R. Moser and W. Grossmann, “Digital twins of complex projects,” in *The Digital Twin*, N. Crespi, A. T. Drobot, and R. Minerva, Eds., Cham: Springer, 2023, pp. 677–702.
- [87] F. Barhebwa-Mushamuka and S. Wagner, “Multi-partners digital project twin: A tool for project monitoring,” *IFAC-PapersOnLine*, vol. 55, no. 10, pp. 383–388, 2022.
- [88] E. F. Shakirov, N. Kattner, C. Fortin, I. K. Uzhinsky, and U. Lindemann, “Reducing the uncertainty in engineering change management using historical data and simulation modelling: A process twin concept,” *International Journal of Product Lifecycle Management*, vol. 13, no. 1, pp. 89–114, 2021.
- [89] A. Maria, “Introduction to modeling and simulation,” *Proceedings of the 29th Conference on Winter Simulation*, pp. 7–13, 1997.
- [90] R. Atkinson, “Project management: Cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria,” *International Journal of Project Management*, vol. 17, no. 6, pp. 337–342, 1999.
- [91] M. A. Cohen, J. Eliasberg, and T.-H. Ho, “New product development: The performance and time-to-market tradeoff,” *Management Science*, vol. 42, no. 2, pp. 173–186, 1996.
- [92] B. L. Bayus, “Speed-to-market and new product performance trade-offs,” *Journal of Product Innovation Management*, vol. 14, no. 6, pp. 485–497, 1997.
- [93] J. P. Lewis, *Project Planning, Scheduling, and Control: The Ultimate Hands-On Guide to Bringing Projects in On Time and On Budget*. New York: McGraw-Hill Education, 2011, vol. 5.
- [94] T. R. Browning and S. D. Eppinger, “Modeling impacts of process architecture on cost and schedule risk in product development,” *IEEE Transactions on Engineering Management*, vol. 49, no. 4, pp. 428–442, 2002.

- [95] T. R. Browning, "A quantitative framework for managing project value, risk, and opportunity," *IEEE Transactions on Engineering Management*, vol. 61, no. 4, pp. 583–598, 2014.
- [96] A. A. Yassine, W. Daniel E, and T. Zambito, "Assessment of rework probabilities for simulating product development processes using the design structure matrix (DSM)," *International Design Engineering Technical Conferences*, pp. 105–113, 2001.
- [97] S.-H. Cho and S. D. Eppinger, "A simulation-based process model for managing complex design projects," *IEEE Transactions on Engineering Management*, vol. 52, no. 3, pp. 316–328, 2005.
- [98] S. Gallego-García, D. Ren, D. Gallego-García, S. Pérez-García, and M. García-García, "Dynamic innovation information system (diis) for a new management age," *Applied Science*, vol. 12, no. 6592, 2022.
- [99] A. M. Anderson, "A framework for npd management: Doing the right things, doing them right, and measuring the results," *Trends in Food Science & Technology*, vol. 19, pp. 553–561, 2008.
- [100] M. V. Tatikonda, "Product development performance measurement," in *Handbook of New Product Development Management*, C. H. Loch and S. Kavadias, Eds., New York, NY: Taylor & Francis, 2008, pp. 199–215.
- [101] M. Hassannezhad, M. Cantamessa, F. Montagna, and P. J. Clarkson, "Managing sociotechnical complexity in engineering design projects," *Journal of Mechanical Design*, vol. 141, 2019.
- [102] R. E. Levitt, "The virtual design team: Designing project organizations as engineers design bridges," *Journal of Organization Design*, vol. 1, no. 2, pp. 14–41, 2012.
- [103] X. Zhang, L. Luo, Y. Yang, Y. Li, C. M. Schlick, and M. Grandt, "A simulation approach for evaluation and improvement of organisational planning in collaborative product development projects," *International Journal of Production Research*, vol. 47, no. 13, pp. 3471–3501, 2009.
- [104] Y. Jin and R. E. Levitt, "The virtual design team: A computational model of project organizations," *Computational & Mathematical Organization Theory*, vol. 2, no. 3, pp. 171–196, 1996.
- [105] D. C. Wynn, N. H. M. Caldwell, and P. J. Clarkson, "Predicting change propagation in complex design workflows," *Journal of Mechanical Design*, vol. 136, 2014.

- [106] X. Zhang and V. Thomson, “Modelling the development of complex products using a knowledge perspective,” *Research in Engineering Design*, vol. 30, pp. 203–226, 2019.
- [107] M. Grimaldi, M. Greco, L. Cricelli, F. Rogo, and M. Scalvenzi, “How can productivity in product design and engineering be assessed? Guidelines to build a dashboard of KPIs,” *IEEE Transactions on Engineering Management*, vol. 71, pp. 560–573, 2024.
- [108] S. A. Piccolo, S. Lehmann, and A. M. Maier, “Different networks for different purposes: A network science perspective on collaboration and communication in an engineering design project,” *Computers in Industry*, vol. 142, 2022.
- [109] H. Jaber, F. Marle, and M. Jankovic, “Improving collaborative decision-making in new product development projects using clustering algorithms,” *IEEE Transactions on Engineering Management*, vol. 62, no. 4, pp. 475–483, 2015.
- [110] J. M. Colombi, M. P. Kretser, J. Ogden, and P. Hartman, “Enterprise systems integration using collapsing design structure matrices,” *CrossTalk*, vol. 29, no. 3, pp. 33–37, 2016.
- [111] I. de Fontaines, D. Lefeuvre, G. Prudhomme, and M. Tollenaere, “Improving digital engineering tools in complex product development by means of an adequate monitoring of research projects,” *IFIP International Federation for Information Processing*, pp. 210–219, 2013.
- [112] T. McDermott, P. Collopy, M. Nadolski, and C. Paredis, “The future exchange of digital engineering data and models: An enterprise systems analysis,” *Procedia Computer Science*, vol. 153, pp. 260–267, 2019.
- [113] A. Yassine, N. Joglekar, D. Braha, S. Eppinger, and D. Whitney, “Information hiding in product development: The design churn effect,” *Research in Engineering Design*, vol. 14, pp. 145–161, 2003.
- [114] A. Borshchev and A. Filippov, “From system dynamics and discrete event to practical agent based modeling: Reasons, techniques, tools,” *22nd International Conference of the System Dynamics Society*, 2004.
- [115] V. Lévardy and T. R. Browning, “An adaptive process model to support product development project management,” *IEEE Transactions on Engineering Management*, vol. 56, no. 4, pp. 600–620, 2009.
- [116] K. G. Cooper, “Naval ship production: A claim settled and a framework built,” *Interfaces*, vol. 1, no. 6, pp. 20–36, 1980.

- [117] J. M. Lyneis and D. N. Ford, "System dynamics applied to project management: A survey, assessment, and directions for future research," *System Dynamics Review*, vol. 23, no. 2/3, pp. 157–189, 2007.
- [118] D. Kasperek, S. Maisenbacher, and M. Maurer, "Structure-based analysis of dynamic engineering process behavior," *2014 IEEE International Systems Conference Proceedings*, pp. 83–88, 2014.
- [119] F. Yin, Q. Gao, and J. Sun, "Research and application of the simulation method for product development process based on system dynamics," *Systems*, vol. 12, no. 172, pp. 171–196, 2024.
- [120] C. L. Storto, G. D'Avino, P. Dondo, and V. Zezza, "Simulating information ambiguity during new product development: A forecasting model using system dynamics," *International Journal of Modelling, Identification and Control*, vol. 3, no. 1, pp. 97–110, 2008.
- [121] S. A. Kauffman and E. D. Weinberger, "The NK model of rugged fitness landscapes and its application to maturation of the immune response," *Journal of Theoretical Biology*, vol. 141, no. 2, pp. 211–245, 1989.
- [122] M. J. Songhori, M. Tavana, and T. Terano, "Product development team formation: Effects of organizational- and product-related factors," *Computational and Mathematical Organization Theory*, vol. 26, pp. 88–122, 2020.
- [123] F. Wöhr, S. Königs, P. Ring, and M. Zimmermann, "A role-activity-product model to simulate distributed design processes," *Proceedings of the 22nd International DSM Conference*, 2020.
- [124] P. Dutta, CarlaPepe, and H. Xi, "Analyzing organization structures and performance through agent-based socio-technical modeling," *Proc. of the 18th Asia Pacific Symp. on Intell. & Evol. Systems*, vol. 2, pp. 39–53, 2015.
- [125] S. Duckwitz, S. Tackenberg, S. Karahancer, and C. M. Schlick, "A meta-model for actor-oriented, person-centered simulation for the management of development projects," *IEEE 17th International Conference on Industrial Engineering and Engineering Management*, pp. 473–477, 2010.
- [126] T. Licht, L. Schmidt, C. Schlick, L. Dohmen, and H. Luczak, "Simulation-based multiple project management in engineering design," *IEEE Transactions on Engineering Management*, pp. 543–554, 2007.

- [127] R. M. Crowder, M. A. Robinson, H. P. N. Hughes, and Y.-W. Sim, “The development of an agent-based modeling framework for simulating engineering team work,” *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans*, vol. 42, no. 6, pp. 1425–1439, 2012.
- [128] K. Tahera, C. Earl, and C. Eckert, “A method for improving overlapping of testing and design,” *IEEE Transactions on Engineering Management*, vol. 64, no. 2, pp. 179–192, 2017.
- [129] P. V. Norden, “Project life cycle modelling: Background and application of the life cycle curves,” *DTIC ADA053014: Software Phenomenology - Working Papers of the Software Life Cycle Management Workshop*, pp. 217–312, 1977, URL: www.archive.org/details/DTIC_ADA053014/page/217.
- [130] H. Van der Auweraer and J. Leuridan, “A new testing paradigm for today’s product development process - part 1,” *Sound and Vibration*, pp. 14–18, 2005.
- [131] D. J. Wagg, K. Worden, R. J. Barthorpe, and P. Gardner, “Digital twins: State-of-the-art and future directions for modelling and simulation in engineering dynamics applications,” *ASME Journal of Risk Uncertainty Part B*, vol. 6, no. 3, 2020.
- [132] R. Nattermann and R. Anderl, “The W-model using systems engineering for adaptronics,” *Procedia Computer Science*, vol. 16, pp. 937–946, 2013.
- [133] W. L. Oberkampf, “Simulation accuracy, uncertainty, and predictive capability: A physical sciences perspective,” in *Computer Simulation Validation: Fundamental Concepts, Methodological Frameworks, and Philosophical Perspectives*, C. Beisbart and N. J. Saam, Eds., Cham: Springer, 2019, pp. 69–97.
- [134] S. R. Mueller *et al.*, “Towards a methodology for the definition and evaluation of enterprises through modeling and simulation: Application to product development,” *AIAA Scitech Forum*, 2025.
- [135] A. G. Loerch, “Learning curves,” in *Encyclopedia of Operations Research and Management Science*, S. I. Gass and M. C. Fu, Eds., Boston, MA: Springer US, 2013, pp. 871–874.
- [136] S. Suss and V. Thomson, “Optimal design processes under uncertainty and reciprocal dependency,” *Journal of Engineering Design*, vol. 23, no. 10-11, pp. 829–851, 2012.
- [137] T. Allen, “Architecture and communication among product development engineers,” *California Management Review*, vol. 49, no. 2, pp. 23–41, 2007.
- [138] JMP[®], Student Edition 18. SAS Institute Inc., Cary, NC, 1989–2025.

- [139] M. McKay, R. Beckman, and W. Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, no. 2, pp. 239–245, 1979.
- [140] G. E. Box and D. W. Behnken, "Some new three level designs for the study of quantitative variables," *Technometrics*, vol. 2, no. 4, pp. 455–475, 1960.
- [141] W. Kerley and P. Clarkson, "Modelling the quality of engineering designs," in *Modelling and Management of Engineering Processes*, P. H. . P. J. C. . S. Vajna, Ed., London: Springer, 2010, pp. 153–164.
- [142] J. Brock, S. von Enzberg, A. Kühn, and R. Dumitrescu, "Towards business use cases of process mining in product development and manufacturing: Deriving and classifying 18 application scenarios in industry," *Procedia CIRP*, vol. 128, pp. 268–273, 2024.
- [143] E. Rigger, T. Vosgien, S. Bitrus, P. Szabo, and B. Eynard, "Enterprise architecture method for continuous improvement of plm based on process mining," *Product Lifecycle Management Enabling Smart X*, vol. 594, pp. 563–575, 2020.
- [144] C. Meier, A. A. Yassine, T. R. Browning, and U. Walter, "Optimizing time–cost trade-offs in product development projects with a multi-objective evolutionary algorithm," *Research in Engineering Design*, vol. 27, pp. 347–366, 2016.
- [145] F. Yin, Q. Gao, and X. Ji, "Simulation and optimization of overlapped iteration process with multi-coupled activities based on value analysis," *Concurrent Engineering: Research and Applications*, vol. 27, no. 1, pp. 57–67, 2019.
- [146] S. Kherbachi, Q. Yang, and S. Z. Khan, "A structured approach to measuring and optimizing the organizational architecture in global product development projects," *Concurrent Engineering: Research and Applications*, vol. 28, no. 3, pp. 161–174, 2019.