University of Amsterdam

Masters Thesis

From Milestones to Forecasts: A Digital Shadow for Program Execution at the Rijksvastgoedbedrijf

Author:
Sven Haarbrink

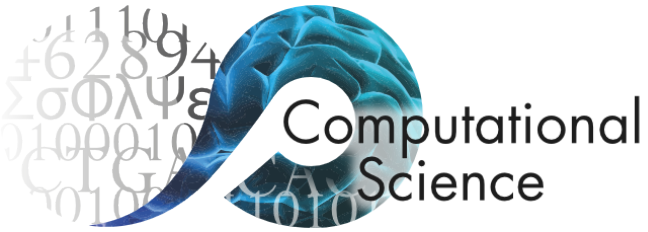
Examiner:
Dr. A.S.Z. Belloum
Supervisor:
Dr. M.J.C. Wilhelm
Assessor:
Dr. Z. Zhao

A thesis submitted in partial fulfilment of the requirements for the degree of Master of Science in Computational Science

in the

Computational Science Lab
Informatics Institute

June 2026


Computational
Science

Declaration of Authorship

I, Sven Haarbrink, declare that this thesis, entitled 'From Milestones to Forecasts: A Digital Shadow for Program Execution at the Rijksvastgoedbedrijf' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at the University of Amsterdam.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date: 26 June 2026

"What I cannot create, I do not understand."

Richard P. Feynman

UNIVERSITY OF AMSTERDAM

Abstract

Faculty of Science
Informatics Institute

Master of Science in Computational Science

From Milestones to Forecasts: A Digital Shadow for Program Execution at the Rijksvastgoedbedrijf

by Sven Haarbrink

The Rijksvastgoedbedrijf (RVB) is the organisation that manages the Dutch national government's real estate portfolio, executes much of its work through programs. These programs are bundles of thematic projects that share role-based capacity. Program managers must decide which projects to start, how to anticipate on delays, and how to prioritise work when bottlenecks form, yet these decisions are typically made based on experience rather than by a quantitative understanding of program-level dynamics. This thesis develops a process-oriented digital shadow of RVB program execution to give insight in the program dynamics. The model is implemented as a discrete-event simulation in which projects progress through six standardised milestones (PO to OAB) while competing for finite, role-specific resource pools.The milestone transition durations emerge out of the model from the interaction between project complexity, role-specific effort, queueing, and passive delay. Complexity is quantified through an expert-weighted composite score derived with the Analytic Hierarchy Process, and the effort of each role is sampled conditionally on this score using complexity-dependent quantile regressions fitted to historical hours-registration data. The shadow is calibrated on four RVB datasets covering milestones, building metadata, project–building mappings, and registered hours. Validated against empirical transition distributions, the model reproduces all five milestone transitions with Kolmogorov–Smirnov statistics below $D = 0.06$, three of which pass the formal two-sample KS test at the 5% level. Scenario experiments demonstrate that program execution is dominated by waiting time: passive delay explains far more of the calendar time between milestones than active role effort, although part of this delay reflects active work performed under separate regional sub-project numbers that the available hours data do not capture. Consequently, role utilisation stays low even at five times the historical workload, complexity has only a modest effect on lead time, and the choice of prioritisation rule barely changes outcomes. Finally, the shadow is run forward from a live data snapshot to produce probabilistic completion forecasts and deadline-risk alerts. The results show that improving program predictability at RVB depends less on internal capacity than on the external and handover processes that make projects wait.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Maite Wilhelm, for the guidance, feedback, and support throughout this thesis. I would also like to thank my examiner, Dr. Adam Belloum, and my assessor, Dr. Zhiming Zhao, for taking the time to evaluate my work and for the valuable feedback provided.

Furthermore, I am grateful to the Rijksvastgoedbedrijf for the opportunity to conduct my thesis in a practical and relevant setting. I would like to thank all colleagues who contributed through helpful input, support, and a pleasant working environment during this process.

Finally, I would like to thank my family and loved ones for the encouragement, and support throughout this journey.

Contents

List of Figures

List of Tables

Abbreviations

CSL	**C**omputational **S**cience **L**ab
UvA	**U**niversiteit **v**an **A**msterdam
RVB	**R**ijks**V**astgoed**B**edrijf
IPM	**I**ntegral **P**roject **M**anagement
DES	**D**iscrete-**E**vent **S**imulation
ABS	**A**gent **B**ased **S**imulation
SD	**S**ystem **D**ynamics
BIM	**B**uilding **I**nformation **M**odeling
FTE	**F**ull-**T**ime **E**quivalents
FIFO	**F**irst-**I**n-**F**irst-**O**ut
AHP	**A**nalytic **H**ierarchy **P**rocess
CL	**C**omplexity **L**evel
DTC	**D**igital **T**win **C**onstruction
RCMPSP	**R**esource-**C**onstrained **M**ulti-**P**roject **S**cheduling **P**roblem
KS	**K**olmogorov–**S**mirnov
KDE	**K**ernel **D**ensity **E**stimate
CV	**C**oefficient of **V**ariation
CI	**C**onfidence **I**nterval

Chapter 1

Introduction

The Rijksvastgoedbedrijf (RVB) is increasingly confronted with the challenge of delivering complex bundles of projects, so called programs, under conditions of uncertainty, resource scarcity, and growing societal expectations. The Rijksvastgoedbedrijf is an executive agency that manages the buildings and land of the national government and the Ministry of Defence, and it operates the largest and most diverse real estate portfolio in the Netherlands, including the Caribbean Netherlands. It was formed in 2014 through the merger of four predecessor government real estate services. As the largest real estate organisation in the country, it is responsible for roughly 900 square kilometres of land, around four times the area of the municipality of Amsterdam, and for over 11.5 million m^2 of buildings, maintained and managed by approximately 3,000 employees. This portfolio is exceptionally diverse, ranging from offices, courthouses, and prisons to military airbases, naval facilities, palaces, national museums, and the forests and estates surrounding royal domains. Besides maintaining this portfolio, the organisation is increasingly responsible for carrying out large thematic programs aimed at improving the sustainability of its properties, executing technical replacements, and realising functional modernisation. Because these programs operate across such a heterogeneous portfolio and rely on shared, specialised expertise, their execution is inherently complex to plan and coordinate. A program is considered a bundle of thematic projects. The programs structure their project teams around several types of roles, including management roles, technical roles, stakeholder-oriented roles, contract-related roles, and project control roles. This role-based structure ensures that the main aspects of project execution are distributed across different areas of expertise and addressed in a structured and integrated way within each project team. Even though every project undergoes a recognizable cycle, the performance of every program relies upon specific capacity restrictions and coordination. Specifically, the completion of projects is often accompanied by delays, queuing, and variability.

While there are growing quantities of data related to project-level activity, program-level decisions often seem to be driven by experience, local rules of thumb, and broad planning tools. Program managers must decide which projects to start, how to allocate scarce roles such as project managers and technical advisors, and how to prioritize work when bottlenecks arise. These decisions have a direct impact on lead times, backlog formation, and predictability of outcome. The dynamic and interconnected nature of the program makes it difficult to understand the program-level impact of these kinds of decisions.

This thesis attempts to fill this gap by developing a digital shadow at the program level, which is aimed at studying the execution dynamics of RVB programs. A digital shadow in this context is defined as a computational representation of the essential states, events, resources, and rules that govern program execution. Rather than aiming to reproduce every operational detail, the model focuses on capturing the high-level mechanisms that generate observable outcomes. The digital shadow is aimed to be used as tool to help make data-driven management decisions.

The modeling framework is built on the current structure of RVB's data, in which projects pass through a series of standardized milestones, namely Project Opdracht (PO; Project Assignment), Project Initiatie Document (PID; Project Initiation Document), PID Akkoord Klant (PAK; PID Approval Client), Aanvang Bouw (AB; Start of Construction), Bouwkundige Oplevering (BO; Building Completion), and Overdracht Aan Beheer (OAB; Transfer to Maintenance). Each milestone is associated to a corresponding completion date, allowing for the analysis of time intervals between successive milestones. As part of the digital shadow approach, this milestone framework is used to simulate projects as entities passing through the system within a discrete event simulation (DES) environment. Each project requires the involvement of specific roles, including Project Manager, Construction Engineer, Mechanical Engineer, Electrical Engineer, and Contract Manager. These roles are modeled as limited-capacity resources that perform the required tasks for transitions between milestones. When resource capacity is constrained, queues form and projects are expected to experience delays. In addition, passive waiting times are incorporated to account for dependencies outside program control, such as contractor lead times and approval processes. Figure 1.1 illustrates this milestone-based progression and how role pools are implemented.

The main research question guiding this thesis is: How can a digital shadow be developed to model and understand the execution dynamics of large-scale government real estate programs? To answer this question, the research investigates several sub-questions. First, which data and modeling methods are required to construct a program-level digital shadow that is both interpretable and empirically grounded? Second, how do resource

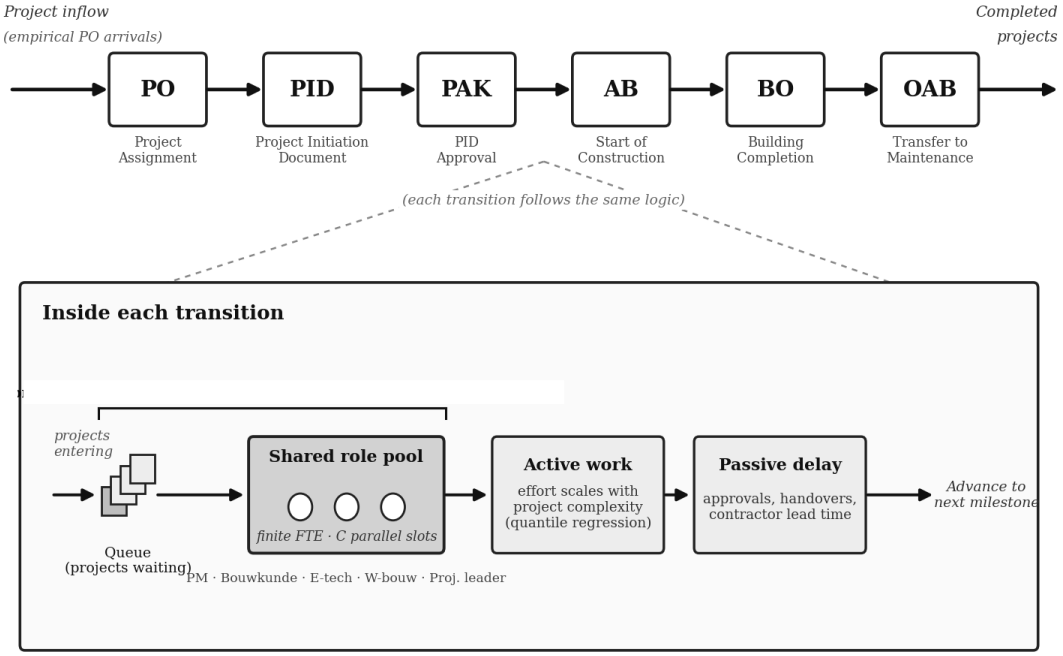


FIGURE 1.1: High-level milestone process flow for RVB program execution. Projects enter at PO and progress through five transitions to OAB; each transition draws on shared, finite-capacity role pools.

allocation and prioritization rules affect program performance metrics such as lead time, backlog, and throughput? Third, how does project complexity influence predictability and variability of outcomes at the program level? Finally, to what extent can execution times be predicted using a combination of simulation and data-driven methods?

To address these questions, the thesis follows a structured approach. A baseline discrete-event simulation model is first developed to represent program execution under shared capacity constraints. This model is then validated against the empirical transition-duration distributions, establishing the data and methods needed for an interpretable, empirically grounded digital shadow. Once validated, it is used for a series of experiments: a complexity sensitivity analysis assesses how complexity affects variability and predictability, while workload and prioritization experiments assess the effect on lead time, backlog, and throughput. Finally, the model is run forward from a live data snapshot to produce probabilistic completion forecasts and deadline-risk alerts.

This thesis contributes to both theory and practice by integrating empirical data, simulation modeling, and uncertainty quantification. Scientifically, this research advances computational method applications to the management of programs in the built environment. Organizationally, it delivers a structured framework to RVB for analyzing execution dynamics, pinpointing bottlenecks, and determining the potential impact of

alternative decision rules before actually deploying them in practice. In summary, the developed digital shadow provides RVB with a decision-support and risk-screening tool, supporting more transparent and data-driven program-level management.

Chapter 2

Literature review

2.1 Conceptual Foundations of Digital Representations

Digital representations have emerged as a key concept in the digital transformation of complex systems. They provide computational representations of physical assets, processes, services, or organisations, and are used to support monitoring, analysis, prediction, and decision-making [1]. However, the terminology used to describe these representations varies across disciplines. Concepts such as digital model, digital shadow, digital twin, process twin, and Digital Twin of an Organization describe related but distinct forms of digital representation.

This section establishes the conceptual foundation for the thesis by distinguishing between two dimensions. The first dimension concerns the *level of integration* between the physical system and its digital representation. This dimension distinguishes digital models, digital shadows, and digital twins according to the direction and automation of data flow [2, 3]. The second dimension concerns the *object of representation*. Digital representations may focus on physical assets, operational processes, services, or organisational systems. These two dimensions are both relevant for this thesis, because the model developed here represents an organisational process.

2.1.1 Levels of Integration: Digital Model, Digital Shadow, and Digital Twin

A common way to classify digital representations is based on the degree of integration between the physical system and the digital representation. In this classification, a *digital model* is a digital representation without automated data exchange. Information from the physical system can be incorporated into the model, but this requires manual

updating. A *digital shadow* extends this by introducing one-way data flow from the physical system to the digital representation. As a result, changes in the physical system can be reflected in the digital representation, but the digital representation does not automatically change or control the physical system. A *digital twin* represents a higher level of integration, in which data flow is bidirectional and changes in the digital representation can also influence the physical system [2, 3].

TABLE 2.1: Levels of integration in digital representations

Concept	**Data flow**	**Interpretation**
Digital model	No automated data flow	A digital representation that is updated manually and does not automatically reflect changes in the physical system.
Digital shadow	One-way data flow from physical to digital system	A digital representation that is updated using data from the physical system, but does not automatically control or change that system.
Digital twin	Two-way data flow between physical and digital system	A digital representation with feedback from the digital system to the physical system, potentially enabling automated intervention or control.

This distinction is important for the terminology used in this thesis. The model developed in this research corresponds to a digital shadow, this positioning is further explained in Section 2.1.3.

2.1.2 Objects of Representation: Assets, Processes, and Organizations

Digital representations can also be distinguished by the type of system they represent. In construction and infrastructure, many applications focus on physical assets, such as buildings, infrastructure components, or construction sites. These asset-oriented representations often build on Building Information Modelling (BIM), sensor data, and performance simulation. Their main purpose is to describe, monitor, or evaluate the state of a physical object [4].

However, digital representations are not limited to physical assets. The literature increasingly recognises that digital representations can also describe processes, services, and organisational systems. In this broader interpretation, the object of representation may be a workflow, a production process, a construction project, or an organisation. This is relevant for process-oriented applications, where the aim is not primarily to represent the geometry or physical condition of an asset, but to understand how work progresses through a system.

In construction, this distinction is reflected in the difference between product information and process information. Product information describes the physical asset, such as geometry, material properties, and building components. Process information describes how the project is executed, including activities, schedules, resources, budgets, and the as-performed state of the work [5]. Similarly, process-oriented digital representations focus on the behaviour of multiple systems or actors within a workflow [2]. This makes them relevant for analysing execution dynamics, resource use, bottlenecks, and delays.

This thesis adopts such a process-oriented perspective. The object of representation is not a single building or physical asset, but the execution process of multiple projects within RVB programs. The model represents projects, milestones, roles, resource capacities, queues, passive delays, and decision rules. In this sense, the thesis is conceptually related to Digital Twin of Organization, because the represented system is organizational rather than purely physical.

2.1.3 Definition for This Thesis

In this thesis, the developed model is defined as a *process-oriented digital shadow* of RVB program execution. The term process-oriented is used because the model represents the execution of projects through milestone transitions, role involvement, resource constraints, queueing, and delays. The term digital shadow is used because the information flow is one-directional. Empirical data from RVB project records, milestone registrations, building data, and hours registration is used to construct, update, and calibrate the simulation model, but the model does not automatically send control actions or decisions back to the physical project system.

This distinction is important in the RVB context. Due to current data infrastructure, hardware, and computational limitations, the model cannot be implemented as a fully integrated, bidirectional digital twin. Instead, it functions as a decision-support model. It enables program managers to analyse execution dynamics, identify bottlenecks, evaluate scenario outcomes, and explore the possible effects of alternative prioritisation or capacity strategies. Interpretation and intervention remain human responsibilities.

The model is therefore positioned as a simulation-based, process-oriented digital shadow. It applies concepts from digital representation literature to the program execution context.

2.2 Digital Twins in Construction and Infrastructure

Digital twin technology is one of the technologies that have attracted a lot of attention in the built environment in recent decades. Most digital twin technology applications have been used to address asset-related digital twins, that is, the BIM-based integration of building performance simulation and Internet of Things data. Duch-Zebrowska and Zielonko-Jung [6] present an instance of the applications of digital twin technology in supporting large-scale retrofitting projects by integrating digital models with simulation tools. In their study, the digital twin is used primarily to enable visualization, performance evaluation, and participatory planning in retrofitting scenarios.

Despite the fact that such approaches identify the strategic importance of digital twins within construction, they remain largely focused on modeling the physical assets. These applications usually focus on energy efficiency, renovation scenarios, and design options without much consideration for the process involved in implementing construction programs of multiple projects.

2.3 Process-Oriented Digital Representations for Project and Program Execution

While digital representation research in the built environment has historically been dominated by asset-oriented applications, more recent work extends the concept towards construction project management, production control, and process-oriented representations. These approaches are relevant because project execution is not only determined by the physical characteristics of assets, but also by planning, coordination, resource availability, decision rules, and delays.

A key contribution in this domain is the concept of *Digital Twin Construction* (DTC) proposed by Sacks et al. [5]. DTC builds on BIM by incorporating construction-site data into production management and feedback processes. The concept is useful for this thesis because it emphasises that construction digital representations should capture not only product information, but also process information, such as tasks, resources, schedules, budgets, and the as-performed state of the project. However, DTC is generally oriented towards construction-site production and closed-loop management, whereas the RVB model developed in this thesis focuses on program-level decision support under one-way data flow.

Simulation methods play an important role in process-oriented digital representations. As highlighted by dos Santos et al. [7], discrete-event simulation and agent-based simulation can serve as analytical cores for dynamic digital representations that are updated using real-world data. Such methods allow managers to anticipate process deviations, analyse system behaviour, and evaluate possible interventions before they are implemented.

Jungmann et al. [8] provide an example of this approach by combining digital representation concepts with location-based management. In their framework, discrete-event simulation is used to analyse what-if scenarios that arise from process deviations on construction sites. By calibrating the simulation model using real-world observations, short-term decision-making can be supported.

Beyond individual projects, the application of process-oriented digital representations to multi-project and program-level environments remains limited. The resource-constrained multi-project scheduling problem has been studied extensively [9], but conventional scheduling approaches often assume deterministic activity durations or fixed sets of projects. Actual program environments are more dynamic. Projects arrive over time, share scarce resources, experience stochastic delays, and interact through queues and capacity constraints.

This thesis addresses this gap by developing a process-oriented digital shadow of RVB program execution. The model is implemented as a discrete-event simulation that is calibrated using empirical RVB data. It represents milestone-based project progression, role-specific resource capacity, project complexity, passive delays, and queueing effects. The model supports forecasting, bottleneck identification, and scenario-based decision support at the program level. Because the model informs human decision-making without automatically controlling the physical project system, it is classified as a digital shadow rather than a full digital twin.

2.4 Project Complexity and Predictability

Kennedy et al. [10] investigate how project complexity shapes the relationship between team communication and project performance, where communication is measured as the self-reported frequency of using face-to-face, telephone, and email contact, and performance covers both technical goal achievement and schedule-and-budget efficiency. Using a Monte Carlo procedure to expand an empirical sample of cross-functional teams, they show that this relationship is curvilinear and that its shape depends on complexity: in highly complex settings, moderate communication can even be detrimental, so that

teams perform best either with minimal communication under a clear strategy or with extensive coordination that fully resolves the uncertainty. Because communication consumes time, they further note that excessive coordination in complex projects translates directly into schedule and budget inefficiency. The relevant implication for this thesis is that complexity does not merely add effort; it changes how coordination affects schedule outcomes, and therefore the predictability of completion times.

Expanding on these foundational concepts, Barani Shikhrobat et al. [11] argue that traditional planning tools such as Gantt charts, CPM, and PERT are fundamentally limited because they rely on deterministic, static assumptions that overlook the dynamic uncertainties inherent in construction environments. They propose a hierarchical framework identifying 20 complexity factors, ranging from externalities and governance to resource availability, and utilize System Dynamics (SD) to model how these factors interact over time.

This work is specifically interesting because of two important reasons. First, it is interesting because it discusses how to use the principle of simulation to create a virtual experiment within a complex environment. Second, it is interesting because it offers a framework to address complex dimension operations beyond simple measures of size or budgets. In a program-level digital shadow framework, complexity exists as a factor that affects the duration between milestone transitions.

Nguyen et al. [12] add to this discussion by suggesting a computational model for the measure of project complexity using a Fuzzy Analytic Hierarchy Process. The authors develop a quantitative Complexity Level (CL) score based on several criteria and subcriteria relevant for construction projects. This structured complexity measure has the potential to bridge qualitative assessments of difficulty and quantitative simulation parameters. In the context of this thesis, complexity scores can be used to define project types or to scale duration and uncertainty parameters in the digital shadow, enabling systematic investigation of how complexity affects lead times and robustness.

Nevertheless, it has been found that a large gap still exists with regards to connecting quantified complex measures directly to dynamic multi-project system models. Specifically, complexity studies typically examine individual projects in isolation, rather than how complex and non-complex projects interact within a shared capacity pool.

2.5 Resource Allocation and Multiproject Dynamics

In the context of multiproject environments, it has been acknowledged that the major determinant of performance does not solely rely on the nature of the projects; rather, it

lies with the allocation of scarce resources among the projects. Discrete event simulation serves as a major environment for the evaluation of the allocation of resources among the projects; the definitions of roles with limited capacity as resources help in the evaluation of the influence of certain priorities. Lu's SDESA framework [13] distinguishes disposable resources from reusable resources with the help of queueing effects.

However, most of the existing literature tends to focus upon the construction site processes or on single-project planning, rather than upon the actual program level governance of public organizations. Additionally, there has been limited research conducted concerning the impact of priority rules at the program level, which include first-in-first-out, urgency-based selection, and the use of the shortest expected duration rule upon the resulting backlogs, throughputs, and predictability of multiple projects. Through the use of priority rules within a digital shadow of the program level, this thesis therefore continues and expands upon the use of resource-constrained simulation within government real estate programs.

Apart from simulation-based approaches, the resource-constrained multi-project scheduling problem (RCMPSP) has been widely studied in the domain of operations research. Villafáñez et al. [14] have discussed the application of a centralized heuristic approach to solve the resource-constrained multi-project scheduling problem. In the discussed approach, the multi-project scheduling problem is considered as a single "mega-project" competing for resources. The constraints of resources have been considered, and the activities have been prioritized with the application of schedule generation schemes and slack based priority rules.

The authors have discussed the centralized and decentralized approaches to multi-project scheduling. In the centralized resource-constrained multi-project scheduling problem, the global objective function is considered as total makespan, and the multi-project scheduling problem is considered as a single entity. The authors explain that the resource-constrained multi-project scheduling problem is computationally very difficult and therefore typically solved using heuristic or metaheuristic approaches instead of exact optimization methods.

While this work offers an important theoretical foundation for understanding the resource competitions between multiple projects, classical RCMPSP models assume deterministic activity duration and a static set of projects known at time zero. In contrast, the projects within RVB programs have stochastic milestone duration, continuous project arrival, and work-in-progress limitations. Thus, despite the formalization of resource sharing and capacity constraints in the RCMPSP literature, the actual execution behavior of the RVB programs differs in some ways from their theoretical model.

2.6 Identified Research Gap and Thesis Positioning

From the reviewed literature, there is significant progress in a few fields of relevance, which include the use of asset-centric digital twins, process-based simulation in construction, calculation of project complexity, and multi-project scheduling under resource constraints. However, the literature from these domains lacks strong interconnectivity. Specifically, there is little work on digital representations which would facilitate the dynamic management of multiple construction projects in light of joint capacity limitations while taking into account variations in project complexity.

The current state of the literature reveals a clear gap. Existing applications of digital representations focus primarily on modelling physical assets and construction processes, rather than the dynamics of program execution. Multi-project scheduling research addresses shared resources, but typically under deterministic conditions and with a static set of projects known in advance. Project complexity research, in turn, provides relevant theoretical and practical metrics, but does not link these to dynamic simulations of how multiple projects interact and jointly consume shared resources.

This thesis positions itself at the intersection of these domains. It develops a discrete-event simulation-based, program-level digital shadow that explicitly models shared role capacities, milestone-based progression, and the interaction between multiple concurrent projects. In addition, it integrates complexity parameters into duration and variability assumptions. In doing so, the thesis aims to provide a computationally informed and empirically grounded framework for analysing execution dynamics, bottlenecks, and predictability in large-scale government real estate programs.

Chapter 3

Methods

3.1 Research Design

In this research, a computer-based data-driven research approach has been adopted to create and investigate a digital shadow for the program execution in RVB. The research aims to create a digital shadow for the dynamic behavior of multi-project programs with shared resource constraints and understand the dynamics of program execution. To do this, data analysis and discrete-event simulation are used, allowing a virtual environment to be created for testing and modeling.

This research distinguishes itself by adopting the digital shadow concept, aiming to develop a reusable model for real-world application rather than a one-time simulation. Therefore, when considering this, the digital shadow, in terms of this research study, means a computer-based interpretation of program execution, similar in structure to actual processes, and periodically updated with actual information from the world. Unlike a regular simulation study, where a one-time analysis is carried out, this study aims to create a model, and by regularly updating information related to program execution, using the information obtained through this model, a digital shadow can be created for analysis and prediction.

The research methodology is designed in a particular research methodological pipeline. To begin with, there is the collection of the dataset, which offers information about the project milestone events, project buildings, and the utilization of the resources. Next, there is the empirical research, which is designed to focus on the critical patterns associated with the execution of the program. In particular, there is the focus on the project milestone transition times, as well as the project arrival characteristics. Next, there is the quantitative complexity model, which is designed to be created based on the

project characteristics, building characteristics, and the differentiation of the projects based on the complexity. Further, there is the creation of the discrete-event simulation model, which is designed to be created based on the insights obtained from the empirical research phase. In the empirical research phase, the projects are represented as entities that move from the milestone state to the next while competing for the available resources in the system. Finally, there are the simulation experiments, which are designed to be conducted based on the created model.

This research design also supports the primary research question on how a digital shadow can be created to understand the execution dynamics of large-scale government real estate programs. The representation of the projects, milestones, resources, and complexity addresses the question on how execution can be translated in a digital shadow. The simulation experiments also help in answering the question on how capacity, allocation rules, and complexity affect execution. The model evaluation against empirical data also helps in answering the question on how well the model captures the dynamics of the past. Finally, the simulation experiments also help in answering the question on how well the model can predict program execution.

The reason for selecting a discrete-event simulation approach is based on the milestone-based data availability and the need to incorporate resource constraints and queue effects. This allows representing the project execution in terms of milestones. The empirical data representation also allows creating a digital shadow that is not limited to analytical experiments.

3.1.1 Data Sources

Four primary datasets are used in this study: a project milestone dataset, a building metadata dataset, a project-to-building mapping dataset, and an hours registration dataset. Each dataset captures a different aspect of program execution. Table 3.1 provides an overview of the datasets, variables extracted, and their role in the modelling process.

Project Milestone Dataset

The project milestone dataset includes data related to the progression of projects through a standard set of milestones. This dataset includes multiple milestone completion dates for each project, related to different stages in the execution of a project. These milestones include, PO, PID, PAK, AB, BO, and OAB.

TABLE 3.1: Overview of data sources, extracted variables, and their role in the modelling process

Dataset	**Main Variables**	**Role in Model**
Project milestone data	Milestone dates, transition durations, project arrival times	Statistical estimation of project flow and processing times
Building metadata	Building size, age, monument status, asbestos indicators	Input for project complexity calculation
Project–building mapping	Project–building relations, number of buildings per project	Aggregation of building attributes to project level
Hours registration	Hours per role, discipline, function name, work date	Statistical estimation of resource demand and role utilization

This dataset acts as the backbone of the digital shadow because it allows the reproduction of the project's trajectory and the estimation of the transition duration between the milestones. The observed milestone dates are not directly used as inputs to the simulation model. Instead, they are used to derive statistical properties of the execution process, such as transition duration distributions and arrival patterns. This ensures that the model is capturing the random nature of the project execution process.

Project–Building Mapping Dataset

The mapping dataset links projects to one or more buildings. This dataset is required because projects in the RVB context are not always associated with a single building, but may involve multiple assets. This dataset was build up by doing interviews with the managers of each program and assigning buildings to each project that falls under their program.

By combining this dataset with the building metadata, representations of building characteristics at the project level can be constructed. This enables the integration of physical asset properties into the modelling of project execution.

Building Metadata Dataset

The building dataset contains information on all buildings managed by the RVB. It includes attributes such as building size, age, and indicators such as monument status or asbestos presence.

This dataset is used to quantify structural characteristics of projects. Since projects can span multiple buildings, these attributes are later aggregated to the project level. The resulting variables serve as inputs for the construction of a project complexity score.

Hours Registration Dataset

The hours registration dataset is a collection of detailed records regarding the time spent on a project by different RVB roles. Each record includes the number of hours worked by a particular role and discipline on a particular project and date. It is important to note that these records cover only the internal RVB effort, such as project management, engineering, and advisory work, and not the physical construction itself. The RVB acts as the commissioning client and does not carry out the building work, that work is performed by external contractors and market parties, whose hours are not part of this dataset. In addition, because the RVB portfolio is large, projects are frequently handed over to a regional team for execution, after which the work is registered under a separate regional sub-project number. The link between a parent project and its regional sub-project is not available in the data used here, so part of the genuine internal role effort cannot be attributed to the originating project.

The information contained in this dataset includes the project, discipline, function name, date, and number of hours worked. It is an empirical tool for understanding the allocation of resources to different projects, the roles that are involved at a particular transition, and the allocation of time over a particular period.

The hours registration data is used to calculate the demand for each role and to analyze the variations that exist regarding the demand for each role, depending on the project and its complexity. This information is later used to inform capacity assumptions and role utilization within the simulation model.

3.1.2 Data Preparation

The raw datasets are processed through several steps to construct a consistent dataset suitable for empirical analysis and simulation modelling.

Data Cleaning and Filtering

The first preprocessing step is to clean the milestone data set. In this step, projects with missing or invalid milestone dates are removed if necessary, to allow for a reliable

computation of transition durations. In addition, date formats are standardized across all data sets to allow for temporal analysis.

The hours registration data set is filtered to only retain relevant data. In this case, valid project identifiers, roles, and hours data are included, while inconsistent data is removed.

Linking Datasets

The data sets are related to each other through the project identifier. The project-building mapping data set is utilized to relate the projects to the relevant buildings. Building-level attributes are then aggregated to the project level, for example by summing or averaging relevant variables.

This step results in a unified dataset in which each project is enriched with both execution data (milestones) and structural characteristics (building attributes).

Construction of Derived Variables

Several derived variables are constructed to enable modelling:

- **Milestone transition durations:** The time between consecutive milestones is calculated using actual completion dates. These durations form the basis for modelling project progression in the simulation.
- **Project arrival times:** The first milestone date (PO) is used to determine the arrival of projects into the system, enabling the modelling of project inflow.
- **Aggregated building characteristics:** Building-level variables such as size, age, and number of buildings are aggregated to the project level.
- **Complexity indicators:** Normalized variables derived from building attributes are used as inputs for the complexity model.
- **Resource demand measures:** Hours registration data is aggregated by project and discipline to estimate the amount of work required per role. These measures are used to inform resource utilization and capacity constraints in the simulation.

Final Modelling Dataset

The result of the preprocessing pipeline is a structured project-level dataset in which each project is characterized by:

- its milestone trajectory and associated transition durations,
- its arrival time into the program,
- its structural characteristics derived from building data,
- and aggregated measures of resource usage based on historical hours registration.

This integrated dataset provides the empirical foundation for the development of the digital shadow. Rather than being used directly as input to the simulation model, it is used to extract statistical patterns and relationships that characterize program execution. These include distributions of milestone transition durations, project arrival processes, complexity indicators, and estimates of resource demand per role.

TABLE 3.2: Example of a single project record in the modelling dataset

Field	**Example value**
project id	47206
program name	Zon op dak
arrival date	2021-03-15
target date	2027-11-11
budget	1250000
bvo (building size)	8450
bouwjaar (building year)	1968
number of buildings	3
monument score	1
asbestos score	0.5
CL_p	6.4

The simulation model is subsequently parameterized using these derived patterns, allowing it to simulate project trajectories of incoming projects based on real-world data. The composite complexity score CL_p shown in the final row of Table 3.2 is defined in Section 3.3.2.

3.2 Empirical Analysis of Program Execution

3.2.1 Milestone Transitions

The milestone dataset forms the primary source for analysing project execution dynamics. Each project progresses through a predefined sequence of milestones, and the time

between these milestones provides insight into the duration and variability of execution phases.

3.2.1.1 Data quality assessment and cleaning

An initial analysis of the milestone dataset revealed substantial data quality issues. A large proportion of milestone date fields contained missing values, with missing percentages exceeding 50% for several milestones. In addition, frequently occurring placeholder dates were identified, most notably in the year 1999, indicating the presence of non-informative entries. For example, a significant number of observations for the final milestone (OAB) were recorded as 1999-01-01.

Further inconsistencies were identified by comparing milestone sequences within projects. In approximately 17.7% of the projects, the recorded completion date of the final milestone (OAB) dated before the start milestone (PO), indicating logically inconsistent data.

To ensure reliable estimation of execution dynamics, the dataset was cleaned by:

- removing placeholder dates (dates in 1999),
- excluding inconsistent project trajectories (e.g., negative lead times),
- retaining only valid "actual" milestone dates for analysis.

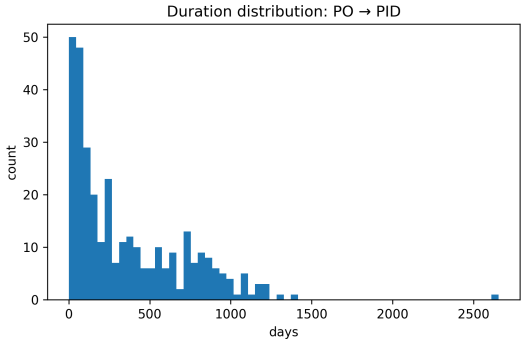
These preprocessing steps ensure data consistency and ensure that the remaining data reflects realistic execution behaviour.

3.2.1.2 Transition duration analysis

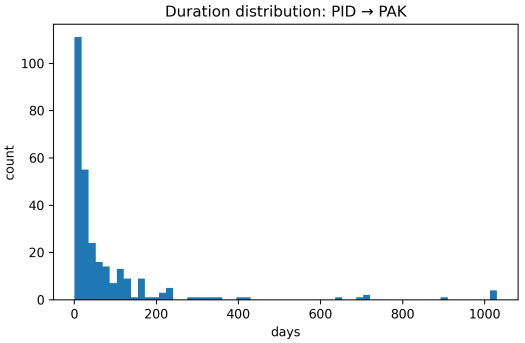
Using the cleaned dataset, transition durations between consecutive milestones were computed. The resulting distributions exhibit substantial variability across all transitions. Table 3.3 summarizes key statistics.

Table 3.3: Summary statistics of milestone transition durations (in days)

Transition	**N**	**Mean**	**Median**	**Std**	**95th pct**
PO → PID	328	360.4	232.5	359.9	998.8
PID → PAK	285	79.8	27.0	161.0	292.6
PAK → AB	305	222.9	124.0	287.5	718.4
AB → BO	220	316.3	215.5	314.1	906.1
BO → OAB	92	434.5	364.5	292.8	884.9

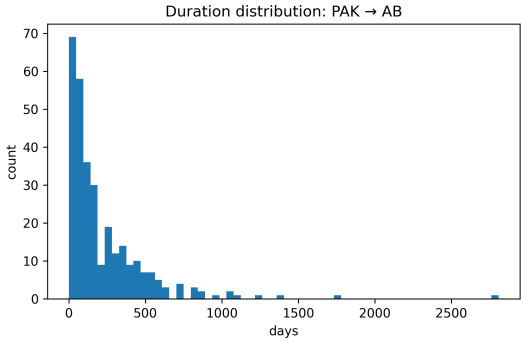


(A) PO → PID

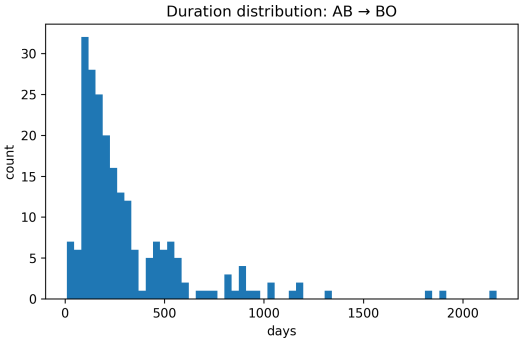


(B) PID → PAK

FIGURE 3.1: Empirical duration distributions for milestone transitions, based on the cleaned milestone dataset.

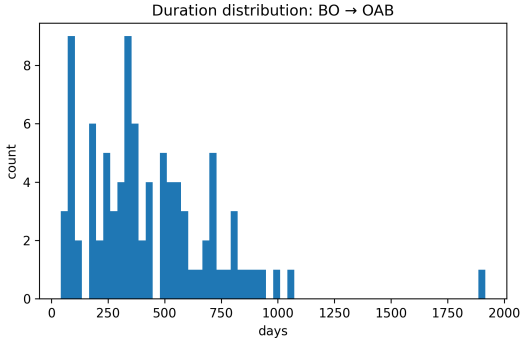


(C) PAK → AB

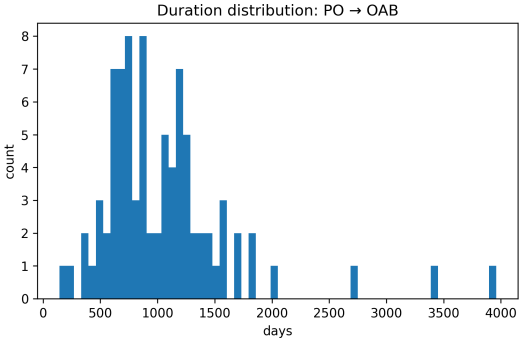


(D) AB → BO

FIGURE 3.1: Empirical duration distributions for milestone transitions, continued.



(E) BO → OAB



(F) PO → OAB

FIGURE 3.1: Empirical duration distributions for milestone transitions and total project lead time, continued.

For all transitions, it can be seen that the mean is greater than the median, implying right-skewed data. Moreover, large standard deviations and high percentile values imply data variability. This suggests that project progression is variable and may depend on the attributes of the project, such as its complexity. These patterns are also visible in the Figure 3.1, where several transitions show a strong concentration of shorter durations combined with a smaller number of very long-running cases.

The total project lead time (PO → OAB) further shows this variability. The average lead time is around 1054 days, with a median of 911 days and a 95th percentile exceeding 1800 days. The wide spread between these values confirms the presence of significant variation in project execution times. As shown in Figure 3.1, this variability is not uniform across transitions: for example, PID → PAK is relatively short and concentrated, whereas PO → PID, PAK → AB, and AB → BO show broader and more dispersed distributions.

A small number of negative duration values are still visible in the transition histograms. This is due to the remaining irregularities in the recorded milestone data on the level of single transitions. While the main cleaning process handled the placeholder dates and the globally inconsistent trajectories (PO to OAB), these local irregularities once more demonstrate the imperfect nature of the operational project data and the need for abstraction before the actual model calibration process.

3.2.1.3 Program-level variation in execution times

In addition to variability across transitions, substantial differences are observed at the level of program groups. Table 3.4 presents summary statistics of total project lead times (PO → OAB) per program.

Table 3.4: Total project lead time in days statistics per program group

Program	**N**	**Mean**	**Median**	**P90**
Legering	111	978	902	1389
Keukens	86	1764	1764	1803
Asbestprogramma	45	535	546	680
XS (TVD)	21	478	478	478
Laadpalen	14	1171	1516	1550
Pachtboerderijen	92	-	-	-

The results show substantial heterogeneity in execution times across programs. For example, projects within the *Keukens* program exhibit average lead times exceeding 1700 days, while programs such as *XS (TVD)* and *Asbestprogramma* show considerably shorter durations. This observation is consistent with the fact that both *XS (TVD)* and *Asbestprogramma* primarily consist of smaller and less complex projects. Table 3.4 also

shows that no lead time statistics are available for *Pachtboerderijen*. This is because the final milestone, OAB, contains the placeholder value 01-01-1999 for all projects in this program. As these placeholder dates were removed during data cleaning, no valid OAB dates remain, preventing the calculation of total lead time statistics.

It is important to note that, in the case of some of the more recent initiated programs, statistics could not be computed because only a few, or no, projects had the PO and OAB milestones recorded. However, projects within these programs often do contain valid timestamps for earlier milestones, meaning that partial transition information remains available for the analysis of intermediate execution phases.

Overall, the differences observed implies that execution dynamics are not homogeneous throughout the portfolio, but rather they depend on program-specific characteristics. This suggests, for instance, that aspects such as scope, availability of roles, building characteristics, or organizational context play a relevant role in determining execution time.

3.2.1.4 Implications for modelling

The empirical results have several implications for the design of the simulation model. First, the variation and right-skewness observed in the transition durations indicate that execution should be modelled stochastically rather than with fixed, deterministic durations. The heterogeneity across projects further suggests that completion times are not uniform but are shaped by project-specific characteristics such as complexity, which motivates linking transition behaviour to project attributes rather than treating all projects identically. Second, the distributions are not only right-skewed but also contain a number of extreme long-duration cases. These long tails are a genuine feature of the operational data and must be preserved in the model; excluding them would underestimate the queueing and congestion that such cases generate under shared capacity. Finally, the substantial differences observed between program groups indicate that project characteristics and program organisation influence execution dynamics. This justifies modelling each program separately, with its own resource configuration, rather than aggregating all projects into a single undifferentiated system.

3.2.2 Arrival Patterns and Project Inflow

Project inflow is defined based on the occurrence of the first milestone (PO), which represents the moment a project enters the execution process. By aggregating PO dates, the temporal pattern of project arrivals can be analysed.

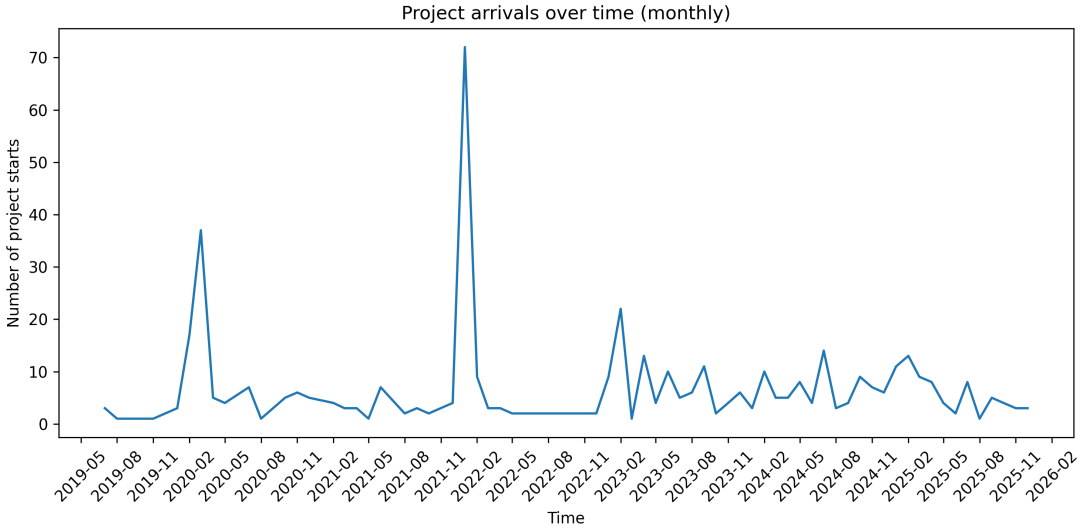


FIGURE 3.2: Number of project starts per quartile based on PO dates.

Figure 3.2 shows that project arrivals vary substantially over time. The inflow is characterised by periods of relatively low activity combined with pronounced peaks, most notably around early 2020 and early 2022. These peaks coincide with the initiation of several programs, such as *Pachtboerderijen* and *Keukens*, which resulted in a spike of project starts during the early phases of program execution.

From the end of 2022, an increase in the capacity of the Rijksvastgoedbedrijf enabled the initiation of additional programs. After this increase, the inflow of projects becomes more regular and follows a more predictable course, with the number of project inflows per month ranging between 5 and 15. This suggests that the inflow becomes more regular once the programs are underway and capacity is sufficient.

Based on these observations, the digital shadow models the inflow as an external process, which is determined by the observed project initiation times. In addition, all the projects are able to enter the system, while the resource constraints control the progress of the projects through the milestones. This allows the representation of the backlog and congestion, which are the consequences of the limited capacity, just as they are in the real world.

3.3 Complexity Modelling

One of the main assumptions of the modeling method applied in this research is that the project complexity affects the efforts and resource consumption involved in the execution of the project. More specifically, higher complexity is associated with greater

required effort and uncertainty, which in turn lead to longer transition durations between milestones. [11, 15]

In this research, complexity is therefore treated as an inherent property of a project that is defined a priori from its budget and building characteristics, rather than being inferred afterwards from observed transition times. The indicators are selected for their domain relevance and their consistent availability at project initiation, and they are combined into a single composite score using expert-derived weights, as described in the following sections.

3.3.1 Complexity Indicators

Project complexity is treated as a latent project attribute that increases workload, coordination needs, and uncertainty, and is therefore expected to influence both the duration and the variability of milestone transitions. Because complexity cannot be observed directly, it is operationalised through a set of measurable project- and building-level indicators. These indicators are combined into a single composite complexity score using weights obtained from domain experts, the weighting procedure is described separately in Section 3.3.2. In the simulation model, this score does not act on transition durations directly but scales the role-specific effort required for each transition (Section 3.3.3.2), so that complexity affects execution time indirectly through increased work content and the resulting exposure to capacity constraints.

3.3.1.1 Indicator definitions

Six indicators are used to characterise project complexity. These indicators are available in the dataset at project initiation, as they primarily describe building-related characteristics.

- **Project budget:** the financial size of the project, used as a proxy for its overall scale and the volume of work involved. Larger budgets are generally associated with larger and more demanding interventions.
- **Number of buildings:** projects involving multiple buildings introduce additional coordination complexity and logistical challenges.
- **Building size (BVO):** larger buildings typically require more extensive interventions and coordination, increasing execution time.

- **Building age:** older buildings are more likely to involve technical uncertainties, outdated infrastructure, or hidden defects.
- **Monument status:** buildings with monument status are subject to stricter regulations and preservation requirements, which can slow down execution.
- **Asbestos presence:** the presence of asbestos introduces additional safety procedures and regulatory constraints, increasing complexity.

3.3.1.2 Aggregation and normalisation

Because a project in the RVB context can be associated with several buildings, the building-level indicators are first aggregated to the project level. The aggregation method for each indicator is summarised in Table 3.5: building size is summed, the construction year is taken as the minimum (i.e. the oldest linked building), the number of buildings is counted, and monument and asbestos status are taken as the maximum observed value across linked buildings, so that the highest risk present in any building drives the project-level score.

TABLE 3.5: Aggregation of building-level indicators to the project level

Indicator	**Aggregation method**	**Interpretation at project level**
Building size (BVO)	Sum	Total floor area associated with the project
Construction year	Minimum	Oldest building linked to the project
Number of buildings	Count	Number of unique buildings linked to the project
Monument status	Maximum	Presence of monument status in at least one building
Asbestos status	Maximum	Highest asbestos-related risk among linked buildings

After aggregation, the indicators are placed on a common $[0, 1]$ scale so that they can be combined into a single weighted score. The four continuous indicators (budget, BVO, building age, and number of buildings) are normalised using a robust min–max scaling based on the 5th and 95th percentiles, as shown in Equation 3.1. This percentile-based scaling limits the influence of extreme outliers, which are common in budget and floor-area data. The two categorical indicators are mapped directly onto $[0, 1]$: monument status is encoded as 0 (no status) or 1 (national monument), and asbestos status as 0 (asbestos-free), 0.5 (clean with restrictions), or 1 (asbestos-bearing).

$$x_{\text{norm}} = \frac{x - Q_{0.05}}{Q_{0.95} - Q_{0.05}}, \qquad x_{\text{norm}} \in [0, 1]. \tag{3.1}$$

3.3.1.3 Empirical relationship with execution duration

To assess whether the selected indicators are relevant for modelling execution dynamics, their relationship with milestone transition durations is examined. Figures 3.3 and 3.4 illustrate the relationship between two representative indicators and the duration of the AB → BO transition.

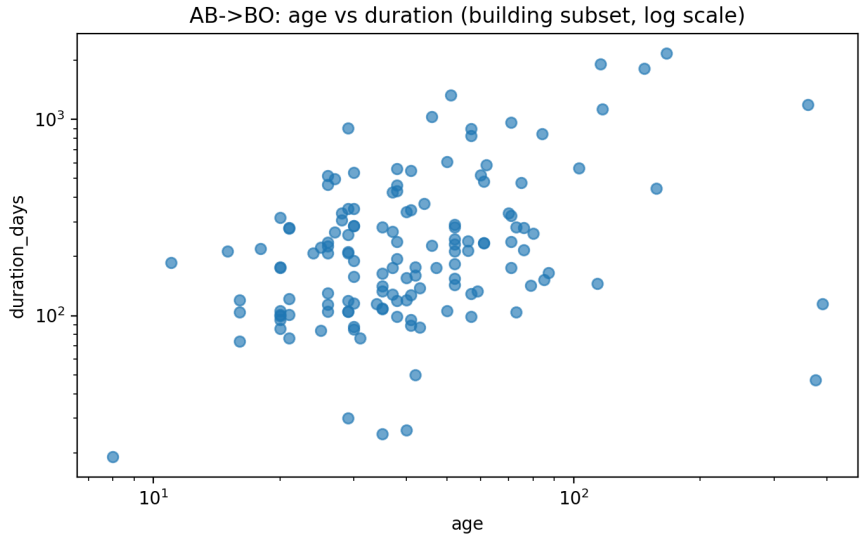


FIGURE 3.3: Relationship between building age and transition duration (AB → BO).

Figure 3.3 shows that, on a logarithmic duration axis, projects involving older buildings tend to shift towards longer execution times, with several long-duration outliers. The trend is positive but noisy: older assets are associated with a higher typical duration and a number of extreme cases, suggesting greater uncertainty and potential delays, although the wide scatter means age alone does not determine duration.

Figure 3.4 indicates that projects with a larger budget tend to have longer durations, although the relationship is not consistent. Larger-budget projects have a higher likelihood of long execution times, which supports the inclusion of budget as one indicator of complexity. Taken together, these observations show that execution time is influenced by both scale-related and asset-related factors, and that no single indicator fully determines complexity. This motivates combining the indicators into a composite measure.

3.3.1.4 Motivation for an expert-weighted composite

The indicators are combined according to three practical requirements. The score must be computable for every incoming project, rely only on consistently available attributes, and remain simple, efficient, and interpretable. The selected indicators, namely project

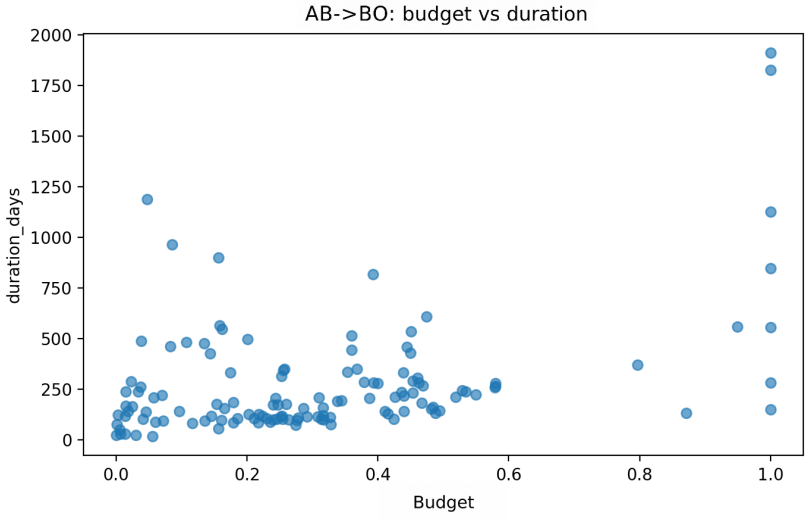


FIGURE 3.4: Relationship between normalised project budget and transition duration (AB → BO).

budget and aggregated building attributes, meet these conditions because they are recorded for nearly all RVB projects. In addition, the indicators should not be weighted equally, since regulatory and preservation constraints, such as monument status or asbestos presence, are expected to influence execution more strongly than attributes such as floor area.

A weighted sum of the normalised indicators satisfies these requirements. It is easy to compute, keeps the indicators explicit, and captures their relative importance through weights. Following Nguyen et al. [12], these weights are obtained from domain experts using the Analytic Hierarchy Process. The next section describes this procedure and the construction of the final complexity score.

3.3.2 Expert-Based Construction of the Complexity Score (AHP)

The relative importance of the six complexity indicators is determined using the Analytic Hierarchy Process (AHP), following the application of AHP to construction project complexity proposed by Nguyen et al. [12]. AHP derives a set of priority weights from pairwise comparisons between criteria, which makes it well suited to settings where the relevant factors are known but their relative importance must be quantified from expert knowledge rather than from data alone.

3.3.2.1 Pairwise comparison questionnaire

A questionnaire was completed by several RVB managers and domain experts, each of whom assessed the relative importance of the six indicators through all $\binom{6}{2} = 15$ pairwise comparisons. For every pair of criteria, the respondent indicated which criterion is more important and by how much, on pre-defined scale ranging from 1 (equal importance) to 9 (extreme dominance of one criterion over the other), with intermediate values 3, 5, and 7. Each response yields an entry a_{ij} of a pairwise comparison matrix, where the reciprocal entry is set automatically: $a_{ji} = 1/a_{ij}$, and $a_{ii} = 1$. This produces, for each expert e, a positive reciprocal matrix $\mathbf{A}^{(e)} \in \mathbb{R}^{6\times 6}$.

3.3.2.2 Weight derivation

For a single comparison matrix $\mathbf{A}$, the priority weights are obtained using the (approximate) eigenvector method: each column is normalised by its sum, and the criterion weight w_i is the average of the normalised entries in row i,

$$w_i = \frac{1}{n}\sum_{j=1}^{n}\frac{a_{ij}}{\sum_{k=1}^{n} a_{kj}}, \qquad \sum_{i=1}^{n} w_i = 1, \tag{3.2}$$

with $n = 6$ criteria. The consistency of the comparisons is assessed through the principal eigenvalue $\lambda_{\max}$, the consistency index (CI), and the consistency ratio (CR):

$$\lambda_{\max} = \frac{1}{n}\sum_{i=1}^{n}\frac{(\mathbf{A}\mathbf{w})_i}{w_i}, \qquad \mathrm{CI} = \frac{\lambda_{\max} - n}{n-1}, \qquad \mathrm{CR} = \frac{\mathrm{CI}}{\mathrm{RI}}, \tag{3.3}$$

where RI is a random consistency index for a matrix of size n (RI = 1.24 for $n = 6$). A consistency ratio below 0.10 is considered acceptable [12].

3.3.2.3 Group aggregation

The individual judgement matrices are combined into a single group matrix through aggregation of individual judgements (AIJ), using the element-wise geometric mean,

$$a_{ij}^{\mathrm{agg}} = \left(\prod_{e=1}^{E} a_{ij}^{(e)}\right)^{1/E} \tag{3.4}$$

after which exact reciprocity ($a_{ji}^{\text{agg}} = 1/a_{ij}^{\text{agg}}$) is restored and the group weights are computed from $\mathbf{A}^{\text{agg}}$ using Equation 3.2. The geometric mean is the standard aggregation operator for AHP because it preserves the reciprocal structure of the comparison matrix, which the arithmetic mean does not.

3.3.2.4 Consistency assessment

Table 3.6 reports the consistency ratio of each individual matrix and of the aggregated matrix. Two experts produced matrices with a consistency ratio above the 0.10 threshold (CR = 0.27 and 0.20), reflecting some internal contradictions in their pairwise judgements, while the remaining were within the acceptable range. Importantly, the aggregated group matrix is highly consistent (CR = 0.025). This improvement is a known property of geometric-mean aggregation: independent inconsistencies in individual judgements tend to partially cancel when combined, so that the group matrix is more coherent than several of its constituents. The aggregated weights are therefore used as the basis for the complexity score.

TABLE 3.6: Consistency ratio (CR) per expert and for the aggregated matrix. A value below 0.10 is considered acceptable.

Source	**Consistency ratio (CR)**
Expert 1	0.268
Expert 2	0.078
Expert 3	0.198
Expert 4	0.076
Expert 5	0.051
Aggregated (geom. mean)	**0.025**

3.3.2.5 Resulting weights and complexity score

The aggregated AHP weights are reported in Table 3.7. Monument status receives the largest weight (0.354), followed by asbestos presence (0.198) and the number of buildings (0.151); building age, budget, and building size receive comparable, smaller weights. This ordering is consistent with the domain interpretation that regulatory and preservation constraints are the dominant drivers of execution difficulty in the RVB portfolio.

The complexity score of a project p is then defined as the weighted sum of its normalised indicators, rescaled to the interval $[0, 10]$:

$$CL_p = 10 \cdot \frac{\sum_{i \in \mathcal{P}_p} w_i \, x_{p,i}}{\sum_{i \in \mathcal{P}_p} w_i}, \tag{3.5}$$

TABLE 3.7: Aggregated AHP weights for the six complexity indicators.

Indicator	**Weight** w_i
Monument status	0.354
Asbestos presence	0.198
Number of buildings	0.151
Building age	0.114
Project budget	0.101
Building size (BVO)	0.082
Total	**1.000**

where $x_{p,i} \in [0, 1]$ is the normalised value of indicator i for project p, w_i is the corresponding AHP weight, and $\mathcal{P}_p$ is the set of indicators that are available (non-missing) for project p. Missing indicators are excluded from both the numerator and the denominator, which redistributes their weight proportionally over the available indicators rather than imputing a value. The division by the available weight keeps the score on a consistent $[0, 10]$ scale regardless of how many indicators are present. Although all indicators should be present for every project this fallback is still implemented to make the model more safe to human error.

The resulting score $CL_p \in [0, 10]$ is the single complexity attribute carried by each project in the simulation model. Section 3.3.3.2 describes how it is used to scale the role-specific effort required for each milestone transition.

3.3.3 Use of the Complexity Score in the Simulation

The complexity score defined in Section 3.3.2 is integrated into the simulation model as a project-specific driver of execution dynamics. Project complexity does not directly determine milestone transition durations. Instead, it influences the amount of transition-specific role effort required, after which transition durations emerge endogenously from the interaction between required work, limited resource capacity, queueing, and passive delays. This subsection describes how the complexity score is translated into model behaviour using the empirical hours registration analysis.

3.3.3.1 Hours registration analysis and transition-specific work content

To translate project complexity into simulation behaviour, a separate empirical analysis was conducted on the hours registration dataset. This analysis was used to identify, for each milestone transition, which roles are typically involved, how frequently they occur, and how much effort they contribute when they are used.

First, the full RVB hours registration data of the last seven years was analysed to identify which roles are most frequently involved across projects. The observed distribution aligns with expectations, as the most common roles reflect the core team structure used across RVB programs. As shown in Figure 3.5, roles such as Werktuigbouwkunde (Mechanical Engineering), Bouwkunde (Building Engineering), Elektrotechniek (Electrical Engineering), and Projectmanagement (Project Management) are the most used in the projects. The role Overige (Other) is an unnamed role used for administrative purposes. A complete mapping of the Dutch source labels to the English labels used throughout this thesis, including the roles shown in Figure 3.5 that are not translated inline, is provided in Appendix A.1 (Table A.1).

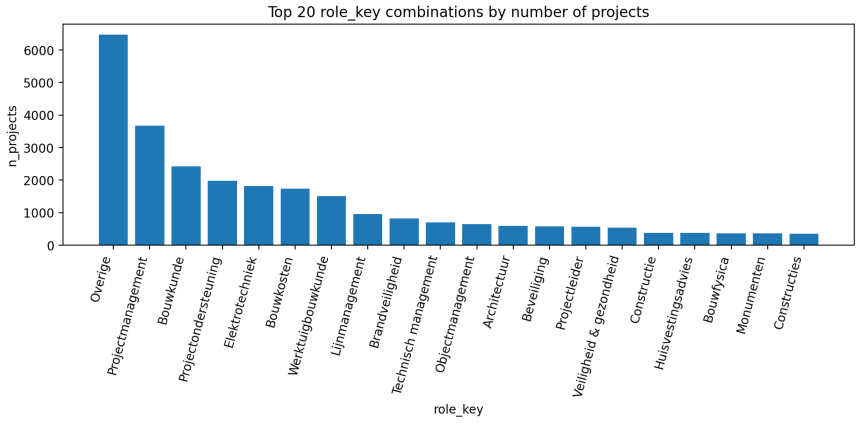


FIGURE 3.5: Frequency of roles used in a RVB projects over the past seven years.

The hours registration data was linked to the project milestone transitions, allowing each registered hour to be assigned to a project milestone transition and role. Based on this, transition specific quantities were derived for each role. First, the empirical prevalence of a role within a transition was used to determine whether that role should be represented in the transition at all. Second, the empirical distribution of registered hours was used to characterize the baseline effort level of that role within the transition. Third, the relationship between observed hours written by a role and project complexity was analysed in order to assess whether more complex projects systematically require more role-specific effort.

This analysis showed that role involvement differs substantially across transitions. Some roles are frequently involved in early project initialization phases, while others become more prominent in technically intensive later phases. In addition, the amount of work performed by a role varies considerably across projects, with right-skewed hour distributions and a limited number of high-effort cases. Finally, the analysis indicates that projects with higher complexity tend to require more effort from key roles, although the strength of this effect differs across roles and transitions. This pattern is visible both at transition level and at role level. For example, Figure 3.6 shows that, for the AB

→ BO transition, projects with higher complexity tend to be associated with higher total written hours, while Figure 3.7 and Figure 3.8 show the same tendency for the Bouwkunde role (Building Engineering) specifically.

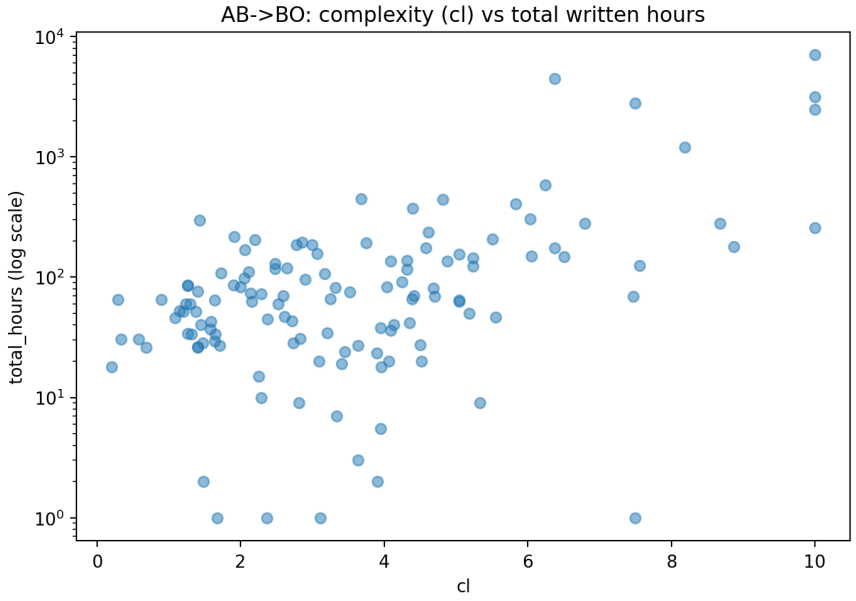


FIGURE 3.6: Observed relationship between project complexity score and total written hours for the AB → BO transition. The vertical axis is shown on a logarithmic scale. Higher-complexity projects tend to exhibit higher total effort requirements and a heavier upper tail.

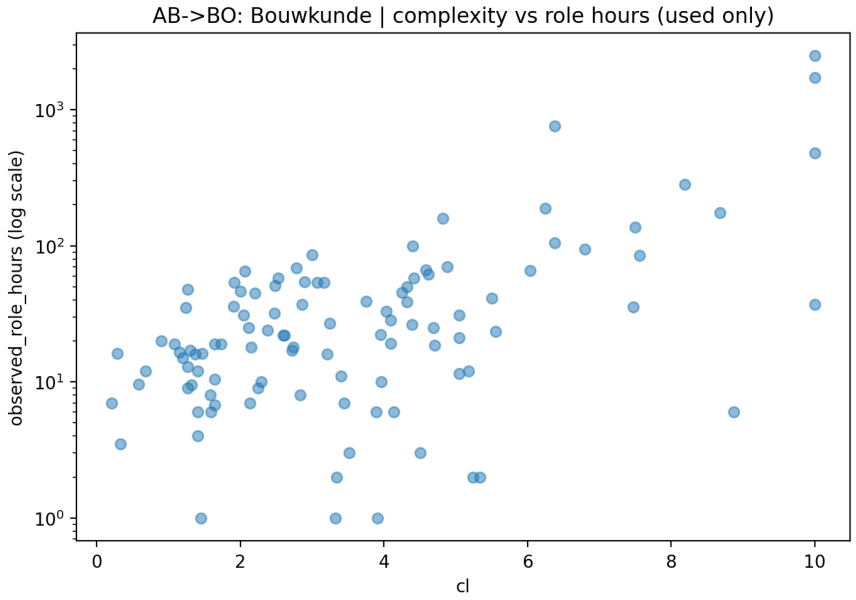


FIGURE 3.7: Observed relationship between project complexity score and Bouwkunde (Building Engineering) role-hours for projects in which the role is active during the AB → BO transition. The vertical axis is shown on a logarithmic scale.

For each transition, a transition-specific work specification is defined containing, for each relevant role, a use rate and the coefficients of two complexity-conditional quantile

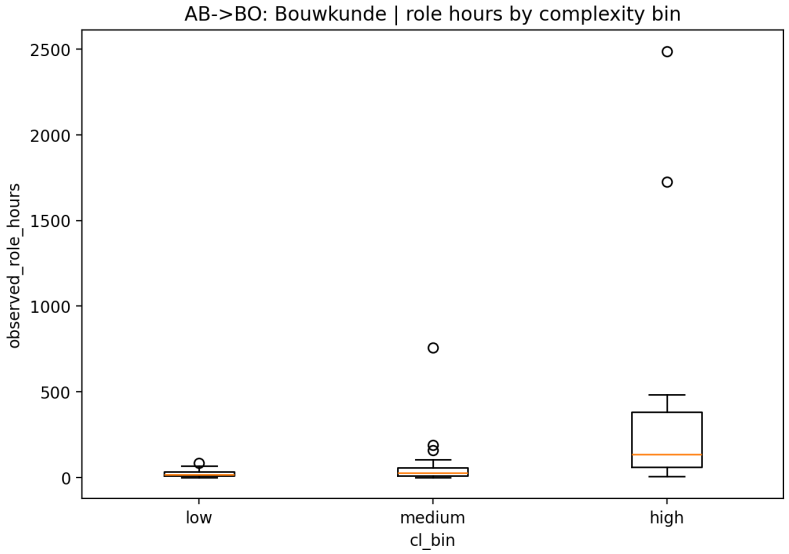


FIGURE 3.8: Distribution of observed Bouwkunde (Building Engineering) role-hours across low-, medium-, and high-complexity projects for the AB → BO transition. Higher complexity bins show both higher central tendency and a broader upper tail.

regressions (for the median and the 90th percentile of the role-hours). The use rate reflects the fraction of historical projects in which the role was active during that transition, while the regression coefficients describe how the required effort of that role scales with project complexity. The full set of parameters used in the simulation model is reported in Table 3.8; the way these coefficients are obtained and used to sample effort is described in Section 3.3.3.2.

3.3.3.2 Translation of complexity into role-specific effort

In the simulation model, project complexity does not directly determine milestone duration. Instead, it scales the amount of work that has to be performed per role, after which the duration emerges from the available capacity, queueing, and passive delays.

Rather than applying a single complexity multiplier to a fixed baseline distribution, the required hours of each role are sampled *conditionally* on the project complexity score using quantile regression. For every role-transition combination, two linear quantile regressions are fitted to the logarithm of the observed role-hours, with the complexity score cl as the single predictor. The first regression targets the median ($\tau = 0.5$) and the second targets the 90th percentile ($\tau = 0.9$):

$$\log m_{r,t}(CL_p) = \beta_{0,r,t}^{(0.5)} + \beta_{1,r,t}^{(0.5)}\, CL_p, \qquad \log q_{r,t}(CL_p) = \beta_{0,r,t}^{(0.9)} + \beta_{1,r,t}^{(0.9)}\, CL_p, \tag{3.6}$$

TABLE 3.8: Quantile-regression parameters per role and milestone transition, for the roles staffed in the simulated programs. For each role the median ($\tau = 0.5$) and 90th-percentile ($\tau = 0.9$) regressions of log(hours) on the complexity score CL_p are reported through their intercept β_0 and slope β_1. n is the number of projects in which the role was active with positive hours, on which the regression for that cell was fitted.

Transition	**Role**	**Use rate**	n	$\beta_0^{(0.5)}$	$\beta_1^{(0.5)}$	$\beta_0^{(0.9)}$	$\beta_1^{(0.9)}$
PO → PID	Project manager	0.383	124	1.081	0.254	2.776	0.249
	Bouwkunde	0.284	92	1.878	0.089	2.473	0.306
	Elektrotechniek	0.182	59	1.288	0.103	2.716	0.106
	Werktuigbouwkunde	0.176	57	1.411	0.121	1.943	0.387
	Project leader	0.083	27	2.333	0.140	3.716	-0.012
PID → PAK	Project manager	0.332	94	1.099	0.078	3.026	0.054
	Bouwkunde	0.198	56	0.803	0.180	1.877	0.335
	Elektrotechniek	0.131	37	0.427	0.189	1.681	0.222
	Werktuigbouwkunde	0.102	29	0.837	0.092	0.747	0.389
	Project leader	0.060	17	2.143	-0.053	3.192	-0.071
PAK → AB	Project manager	0.401	121	1.751	0.217	3.384	0.249
	Bouwkunde	0.397	120	2.589	-0.019	3.143	0.228
	Elektrotechniek	0.334	101	1.624	0.142	2.178	0.378
	Werktuigbouwkunde	0.219	66	1.680	0.175	2.765	0.278
	Project leader	0.119	36	3.021	-0.054	3.905	0.162
AB → BO	Bouwkunde	0.479	105	2.220	0.331	2.854	0.460
	Elektrotechniek	0.416	91	1.841	0.242	2.584	0.436
	Project manager	0.402	88	2.084	0.263	2.893	0.355
	Werktuigbouwkunde	0.320	70	1.801	0.219	2.854	0.312
	Project leader	0.146	32	3.357	0.057	3.976	0.282
BO → OAB	Bouwkunde	0.758	69	2.579	0.043	3.881	-0.095
	Elektrotechniek	0.681	62	2.647	-0.029	3.311	-0.009
	Project manager	0.593	54	1.898	0.163	3.215	0.096
	Werktuigbouwkunde	0.527	48	2.050	0.165	2.829	0.165
	Project leader	0.242	22	3.532	0.043	4.079	0.073

where $m_{r,t}(CL_p)$ and $q_{r,t}(CL_p)$ are the complexity-conditional median and 90th percentile of the required hours for role r in transition t. The logarithm is used because the observed hours are strictly positive and right-skewed, modelling log(hours) keeps the predictions positive and makes the relationship with complexity approximately linear. The median line describes how the typical effort grows with complexity, while the 90th-percentile line describes how the upper tail grows. Fitting both quantiles separately allows the spread of the effort distribution, and not only its centre, to depend on complexity. This is consistent with the empirical pattern in Figures 3.6 to 3.8, where higher-complexity projects show both a higher central effort and a heavier upper tail.

During the simulation, the required hours for an active role are drawn from a lognormal distribution whose median and 90th percentile are obtained by evaluating Equation (3.6) at the project's complexity score:

$$m = e^{\log m_{r,t}(CL_p)}, \qquad q = \max\!\left(e^{\log q_{r,t}(CL_p)},\ 1.05\, m\right), \tag{3.7}$$

$$\sigma = \frac{\log(q/m)}{z_{0.90}}, \qquad \mu = \log m, \qquad h_{r,t} \sim \text{Lognormal}(\mu, \sigma^2), \tag{3.8}$$

with $z_{0.90} = 1.2816$ the standard-normal 90th-percentile quantile and $CL_p \in [0, 10]$ clamped to the range on which the regressions were fitted. Because both m and q depend on cl, both the location and the spread of the sampled effort increase with complexity. The lower bound $q \geq 1.05\, m$ in Equation (3.7) guards against quantile crossing: because the two regressions are fitted independently, the fitted 90th-percentile line can fall below the median line at the extremes of the complexity range, which would otherwise collapse the distribution to a single point. A small floor of 0.25 hour is applied to the sampled effort for numerical stability.

The regression coefficients are estimated only on projects in which the role was actually active and recorded positive hours, so that the parameters reflect the effort *given* that the role is involved, whether a role is involved at all is governed separately by its use rate. The resulting parameters for all modelled roles are reported in Table 3.8.

3.3.3.3 From effort to execution time

The hours sampled for each active role are not processed in a single block. The required effort is divided into work units of at most one working day, and each unit occupies one unit of the role's capacity for one calendar day:

$$h_{chunk} = \min(h_{remaining}, H), \qquad d = \frac{h_{chunk}}{H}, \tag{3.9}$$

where $h_{remaining}$ is the remaining effort for the role, $H = 8$ is the number of working hours in one day, and d is the duration of one work unit in calendar days (equal to one day for all but a possible final partial unit). A role with a capacity of C full-time equivalents can process C such units in parallel on the same day, so that up to C projects can be served simultaneously. When the number of projects requiring a role exceeds its capacity, the additional work units are queued and processed once capacity becomes available.

As a result, the duration of a transition depends not only on the required effort, but also on the available role capacity and the queueing that arises under contention. Complexity therefore influences transition duration indirectly through increased work content, which leads to longer active involvement and greater exposure to capacity constraints and queueing delays.

3.3.3.4 Passive delays

In addition to active role work, each transition includes a passive delay component. This component represents process-related waiting not captured by the direct execution of registered hours, such as decision-making, approvals, handovers, contractor lead times, and coordination gaps. This component also accounts for capacity that is provided by regional teams but is not directly visible in the project-level hour data, because these hours are registered under separate regional sub-project numbers.

Passive delay is estimated as the residual calendar time that remains after subtracting an empirical active-work duration from the observed transition duration. The registered hours of a role are converted into active working days by dividing by the number of working hours in a day, $H = 8$, and the different roles are assumed to work in parallel rather than sequentially. For each project–transition observation i, let H_{ir} denote the total hours registered by role r. The active duration proxy is then the longest role-specific active duration:

$$A_i = \max_r \left(\frac{H_{ir}}{H} \right), \qquad H = 8. \tag{3.10}$$

Using the same hours-to-days conversion as the simulation ensures that the active and passive components are defined consistently across calibration and execution: in both cases a role's effort occupies H_{ir}/H working days, and because roles progress concurrently the active part of a transition is governed by the busiest role rather than by the sum of all role work. The passive delay is then estimated as the remaining observed calendar duration:

$$L_i^{res} = \max\left(0,\, D_i^{obs} - A_i\right), \tag{3.11}$$

where D_i^{obs} is the observed milestone-to-milestone duration of the transition. Observations without registered hours are retained by setting $A_i = 0$, so their full observed duration is attributed to passive delay.

For each transition, the median and 90th percentile of the positive residual delays L_i^{res} are computed empirically and denoted med^{res} and p_{90}^{res}. These two quantiles are converted into the parameters of a lognormal distribution. Following Equation 3.8, the log-scale parameters are calculated using these formulas:

$$\mu_L = \log(\text{med}^{res}), \qquad \sigma_L = \frac{\log(p_{90}^{res}/\text{med}^{res})}{z_{0.90}}, \tag{3.12}$$

with $z_{0.90} = 1.2816$ the standard-normal 90th-percentile quantile. The passive delay for the transition is then drawn as

$$L \sim \text{Lognormal}\big(\mu_L,\, \sigma_L^2\big)\,. \tag{3.13}$$

By construction the sampled distribution reproduces the empirical median and 90th percentile of the residual delays, while its log-normal shape preserves the right-skewed character of the observed waiting times.

The passive delay is added after all active role work for the transition has finished, so the total transition duration is:

$$T = T^{active} + L, \tag{3.14}$$

where T^{active} is the active work duration including any queueing time.

Adding the passive delay at the end of the transition is an approximation, since the waiting may in reality also occur before or during the active work. The assumption is that this simplification is acceptable as long as the role pools are not saturated, the total duration should be independent of the order in which active work and waiting take place.

Table 3.9 summarises the resulting parameters. Passive delays are substantial relative to active work across all transitions, indicating that milestone-to-milestone calendar time is dominated by process waiting rather than registered role effort. The largest absolute passive delays occur in the initiation and later design-to-delivery phases.

TABLE 3.9: Passive delay parameters derived from the role-parallel residual method

Transition	**N**	**Median (days)**	**90th pct. (days)**
PO → PID	324	223.75	861.40
PID → PAK	283	24.00	161.00
PAK → AB	302	114.50	506.90
AB → BO	219	196.00	589.00
BO → OAB	91	345.75	767.00

3.3.3.5 Resulting endogenous transition durations

The combination of the mechanisms above implies that milestone transition durations are not sampled directly from historical transition-time distributions. Instead, transition durations emerge from the interaction between project-specific work content, effort requirements, passive delays, and limited shared resource capacity.

For each project and transition, the final duration is therefore determined by:

- which roles are activated in that transition,
- how many hours each active role requires,
- how the required effort is converted into working days,
- how much role capacity is available,
- how much passive delay occurs within the transition,
- and how long the project must wait for scarce resources to become available.

As a consequence, higher-complexity projects consume more effort, experience more variation in execution, and remain longer in the system. This increases resource competition and queueing, which in turn generates additional congestion effects at program level.

3.4 Digital Shadow Simulation Model

This section describes how program execution is represented in the digital shadow using a discrete-event simulation approach. The model captures how projects progress through milestone phases while competing for limited organizational resources. Rather than prescribing fixed transition durations, execution dynamics emerge from the interaction between project-specific work requirements, resource constraints, and queueing effects.

3.4.1 Model Entities and Project States

The central entity in the simulation model is the *project*. Each project enters the system at its empirically observed arrival time and is initialized using information from the milestone, building, and complexity datasets. Upon entry, a project is assigned a program to which it belongs, a target finish date (milestone OAB), and the project-specific complexity score.

Projects progress through the fixed sequence of milestone states:

$$PO \rightarrow PID \rightarrow PAK \rightarrow AB \rightarrow BO \rightarrow OAB.$$

These states represent the main execution phases observed in the empirical RVB data. At any point in time, a project occupies exactly one state, and progression occurs through transitions between consecutive milestones.

Projects maintain both static and dynamic attributes. Static attributes include project characteristics such as program membership, budget, building properties, and complexity. Dynamic attributes include the current milestone, the timing of milestone entries, and the project's start and completion times. During the simulation, each project records its trajectory, enabling reconstruction of its full execution path.

Advancement between milestone states is not time-driven but process-driven. A project only progresses once the work associated with the current transition has been completed. This means that execution time is not predefined, but emerges from the underlying simulation dynamics.

3.4.2 Resource Constraints and Queues

A key feature of the model is the explicit representation of limited resource capacity. Work is performed by role-specific resource pools, where each role has a finite capacity expressed in full-time equivalents (FTE). Each unit of capacity processes the work of one project at a time in units of one working day, so a role with C FTE can serve up to C projects in parallel on a given day.

In the current baseline implementation, all programs have their own dedicated resource pool. This means that all projects within a program compete for the same role-specific capacity. This program-specific resource structure was derived from interviews with the managers of each program, who provided information on the roles and capacities available within their respective program teams.

When the demand for a role exceeds its available capacity, queues are formed. Each role pool maintains its own queue of pending work requests. As a result, projects do not wait as a whole; instead, individual role-specific work components may experience delays depending on resource availability. This allows different parts of a project to progress at different speeds.

The order in which queued work requests are processed is determined by an allocation policy, which is set at the program level and applied uniformly across all role pools within that program. Three policies are implemented. Under the first-in-first-out (FIFO) policy, the priority key for each request is the project's arrival time t^{arr}, so earlier-arriving projects are always served before later-arriving ones. Under the minimum-slack (MIN-SLK) policy, the priority key is the remaining slack $t^{target} - t^{now}$, where t^{target} is the project's target OAB completion date and t^{now} is the current simulation time. A lower slack value means less time remaining before the deadline, so the most urgent project is always served first. Under the minimum-complexity (MIN-CL) policy, the priority key is

the project's complexity score C_p. Projects with lower complexity values are prioritized before more complex projects, representing a dispatching rule in which simpler projects are processed first. For all three rules, ties are broken by arrival time and then by project identifier.

Queueing dynamics emerge naturally from the imbalance between work demand and limited capacity. When resource utilization increases, queues grow longer, leading to increased waiting times and backlog formation. These effects propagate through the system and are a primary driver of congestion at the program level.

3.4.3 Simulation Events and Project Progression

The simulation works as a discrete-event system in which states transition at certain times when specific events happen. These events consist of the arrival of projects, start and finishing of work, resource availability, and milestone completion. Time progresses from event to event, and this provides an efficient way of modeling the dynamic relationship between the project and the common resource. The general flow of events of a project in the model is presented in Figure 3.9.

Once the project enters a milestone, the model selects the transition to be performed. This enables the model to determine the roles being active by using the transition use rates computed through hours registration as outlined in Section 3.3. The effort required for each active role is sampled conditionally on the project's complexity score using the quantile-regression mechanism of Equations (3.6) to (3.8). This means that the higher the complexity level of a project, the higher the expected effort from the roles' side. Finally, the calculated effort is divided into work units of at most one working day, each occupying one unit of the role's capacity for one calendar day, as in Equation (3.9).

These chunks of work are submitted to their respective role-specific resource pools. In case there is enough capacity in those pools, the chunks of work get executed, otherwise, the chunks wait in the queue until there is enough capacity. Chunks of work get processed in sequence, meaning if once a chunk finishes getting processed, the model checks if there is more work to be done, and in such a case, it initiates processing of another piece.

After all active role-specific work chains for a transition have been completed, the model samples a residual passive delay according to Equation (3.13). This delay represents process-related waiting that remains after active role involvement, such as coordination processes, approval procedures, handovers, contractor lead times, and other dependencies that are not captured as continuously occupied role time.

A transition is completed sequentially: first, all active work must be processed through the role-specific resource pools, including any queueing delays; second, the residual passive delay must elapse. Once both components have occurred, the project advances to the next milestone state, after which the same process is repeated. If the final milestone has been reached, the project exits the system and its trajectory is recorded.

Through this event-driven mechanism, milestone transition durations are not imposed directly, but emerge endogenously from the interaction between project-specific work content, resource constraints, and system dynamics. As illustrated in Figure 3.9, delays arise both from limited capacity (queueing) and from passive process components, allowing the model to capture realistic congestion effects and interdependencies within multi-project program execution.

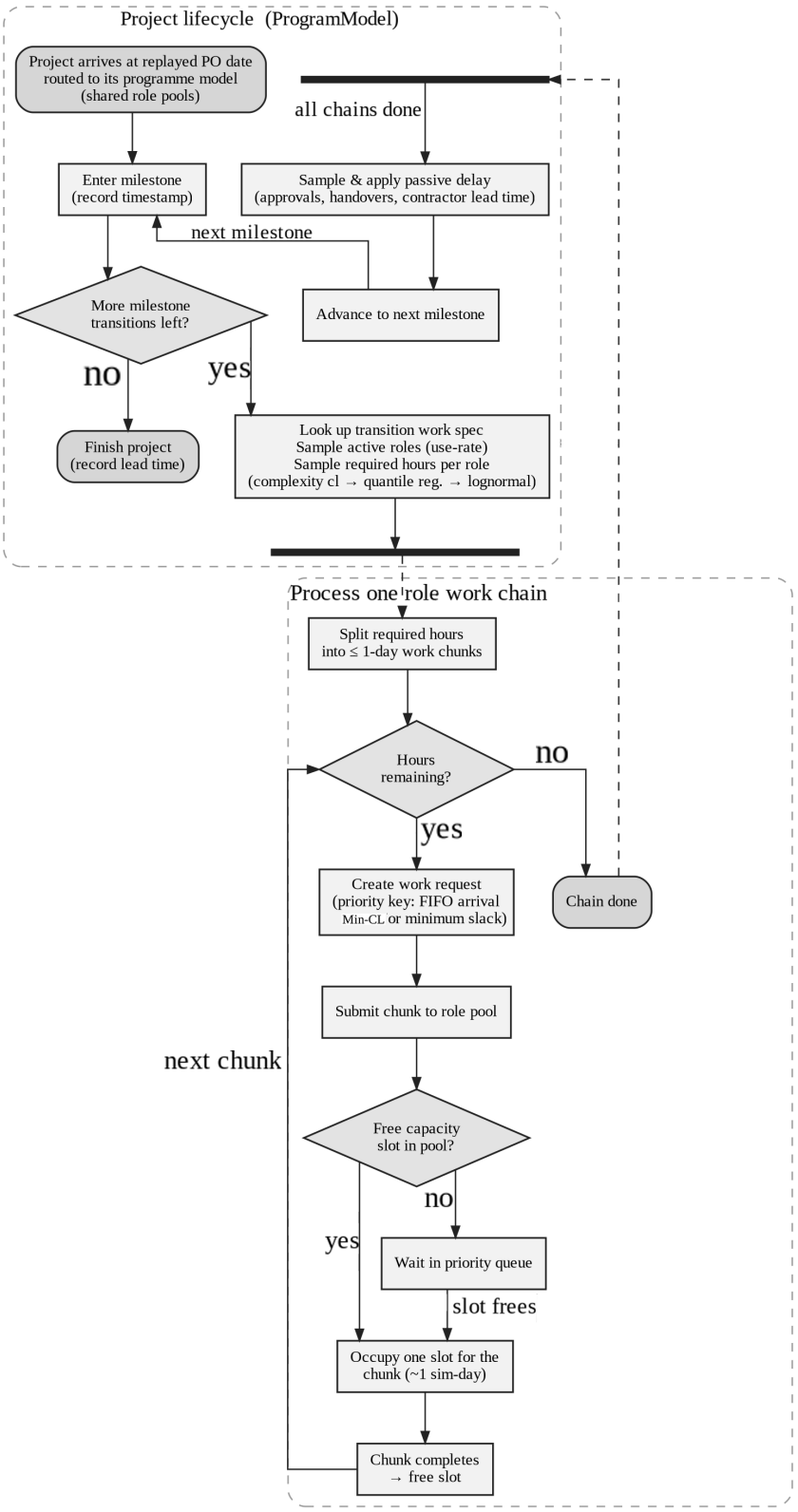


FIGURE 3.9: The transition logic of the simulation model. Visualizing how a project progresses from milestone to milestone.

Chapter 4

Experiments and Results

This chapter presents the experimental evaluation of the digital shadow developed in Chapter 3. The experiments are structured in four stages. First, the baseline model is validated against empirical data to assess whether it reproduces realistic program execution dynamics (Section 4.2). Second, a sensitivity analysis investigates how project complexity affects execution variability and predictability at program level (Section 4.3). Third, a set of scenario experiments evaluates the effect of workload on resource utilisation and lead time, and a second set evaluates the effect of prioritisation rules on program performance (Section 4.4).

4.1 Experimental Setup

This section describes the technical configuration of all the experiments: the simulation engine and replication strategy, the input parameters and baseline resource configuration, and the performance metrics recorded.

4.1.1 Simulation Engine and Replication Strategy

The digital shadow is implemented in Python 3 using a custom discrete-event simulation (DES) engine. The engine maintains a priority-ordered event queue and advances simulation time from event to event, representing project arrivals, work-chunk submissions, resource completions, passive delay expirations, and milestone transitions. The engine is purpose-built to support the blocked work-chunk model and role-pool architecture described in Chapter 3.

Each experiment is run as $N = 30$ independent replications. Each replication uses a distinct random seed drawn from the sequence $\{245, 246, \ldots, 274\}$. Variation enters the model through three independent sampling operations per project-transition:

1. **Role activation** — a Bernoulli draw with the empirically estimated use rate (fraction of historical projects in which the role was active for that transition).
2. **Role effort** — a lognormal draw whose median and 90th percentile are obtained by evaluating the quantile-regression lines (Equations (3.6) to (3.8)) at the project's AHP-weighted complexity score cl.
3. **Passive delay** — a lognormal draw parameterised by the median and 90th percentile derived from the role-parallel residual method (Section 3.3.3.4).

All other inputs, project arrival times, complexity scores, and role capacities, are fixed across replications.

Projects enter the system at their empirically observed PO dates, so the simulation begins with an empty system and loads progressively as historical arrivals replay.

4.1.2 Input Configuration and Baseline Parameters

Project arrival times are replayed directly from the empirical PO dates. Some programs contained projects with inconsistent milestone records, for example placeholder dates in the final OAB milestone. These projects were excluded during preprocessing, which caused some programs to have too few valid projects remaining and therefore to be excluded from the simulation experiments. Each of the six programs runs as an independent simulation with its own set of role-specific resource pools. The baseline staffing configuration for each program is derived from actual capacity data provided by RVB program managers, and is summarised in Table 4.1. Only roles with non-zero capacity in at least one program are listed.

TABLE 4.1: Baseline resource capacity (FTE per role per programme). Only roles with non-zero capacity in at least one program are shown.

Role	**Laadp.**	**Boeg.**	**EML**	**Zon**	**Keuk.**	**XS**
program manager	1	1	1	1	1	1
Bouwkunde advisor	0	0	0	0	0	1
Elektrotechniek adv.	1	0	0	1	0	0
Werktuigbouwkunde adv.	0	0	1	0	1	1
Project leader	1	3	1	5	3	1

Each FTE can serve one project at a time in units of one working day (Section 3.3.3.2), so a role with C FTE can process up to C projects in parallel.

All effort parameters are drawn from Table 3.8 and all passive delay parameters from Table 3.9. The on-time delivery threshold is set to 1,825 calendar days (five years), reflecting the typical program planning horizon used by RVB.

4.1.3 Performance Metrics

The following metrics are recorded for each replication and aggregated across $N = 30$ replications as mean $\pm$ 95% confidence interval using the Student t-distribution with $N - 1$ degrees of freedom.

Project lead time. The total calendar duration from PO to OAB for each completed project, expressed in days. The primary summary statistics reported are the mean, median, standard deviation, and 90th percentile.

Per-transition duration. The calendar duration between consecutive milestones, collected separately for each of the five transitions (PO→PID, PID→PAK, PAK→AB, AB→BO, BO→OAB). Used for model validation in Section 4.2.

Coefficient of variation (CV). The ratio of standard deviation to mean for project lead time ($\text{CV} = \sigma/\mu$). A lower CV indicates a more predictable program.

Tail risk ratio. The ratio of the 90th percentile to the median (P_{90}/P_{50}). A value close to one indicates a compact distribution; a large ratio indicates a heavy upper tail and high tail risk.

Role utilisation. The fraction of available capacity actually in use, computed as:

$$U_r = \frac{\text{total busy time}_r}{C_r \cdot T_{\text{sim}}}, \tag{4.1}$$

where C_r is the FTE capacity of role r and T_{sim} is the total simulation duration. A value near zero indicates idle capacity; a value near one indicates saturation.

Time-average queue length. The time-weighted mean number of work requests waiting in each role pool, approximated from a snapshot history of queue-length observations recorded at each arrival and departure event.

Throughput. The number of projects completing the OAB milestone per calendar year, computed as the count of completions divided by the span from first project start to last project finish.

On-time delivery rate. The fraction of completed projects whose lead time falls within the 1,825-day threshold, reported as a percentage.

4.2 Baseline Model Validation

Before conducting scenario experiments, the digital shadow is validated against historical program execution data. Validation is performed by running $N = 30$ replications with baseline capacity and the observed project arrival times, and comparing the resulting simulated distributions to the empirical statistics derived from the cleaned milestone dataset described in Section 3.2. Validation is assessed along four dimensions: per-transition duration distributions, total project lead time, program-level lead time heterogeneity, and role utilisation.

4.2.1 Transition Duration Distributions

Table 4.2 reports summary statistics for the empirical and simulated duration distributions for each of the five milestone transitions, together with the two-sample Kolmogorov–Smirnov (KS) statistic D. The simulated sample contains $n = 4{,}500$ observations per transition (150 historical projects distributed across six programs, times 30 replications). Because of the large simulated sample size, the p-value is sensitive to even small distributional differences. Therefore, the D statistic, which measures the maximum distance between the empirical and simulated cumulative distributions, is used as the main indicator of practical model fit. Values below 0.10 indicate distributions that are effectively indistinguishable in practice.

TABLE 4.2: Empirical and simulated summary statistics for milestone transition durations (days) across $N = 30$ replications, together with the two-sample KS statistic D and its result at the $\alpha = 0.05$ level.

	Empirical				**Simulated**				
Transition	**Mean**	**Med.**	**Std**	**P90**	**Mean**	**Med.**	**Std**	**P90**	**KS** D
PO → PID	360.4	232.5	359.9	998.8	419.8	243.5	644.4	934.6	0.055
PID → PAK	79.8	27.0	161.0	292.6	75.1	29.0	190.5	164.8	0.048
PAK → AB	222.9	124.0	287.5	718.4	237.9	128.6	343.9	544.9	**0.038***
AB → BO	316.3	215.5	314.1	906.1	311.9	212.8	340.8	639.0	**0.020***
BO → OAB	434.5	364.5	292.8	884.9	440.2	364.2	288.5	812.6	**0.045***

* Passes the two-sample KS test at $\alpha = 0.05$ ($p > 0.05$).

All five transitions produce KS statistics below $D = 0.06$, which is a strong result for a complex discrete-event simulation model. Three transitions pass the KS test formally: PAK→AB ($D = 0.038$, $p = 0.184$), AB→BO ($D = 0.020$, $p = 0.881$), and BO→OAB ($D = 0.045$, $p = 0.072$). The remaining two fail marginally, with PID→PAK barely below the significance threshold ($p = 0.047$) and PO→PID at $p = 0.015$.

The two later transitions show the closest agreement. **AB→BO** achieves the best fit of all five transitions ($D = 0.020$), with a simulated median of 212.8 days against an

empirical median of 215.5 days (error: −1.3%) and a mean error of −1.4%. The standard deviation ratio of 1.09 indicates nearly identical spread. The KDE in Figure 4.4 shows the two curves overlapping substantially across the full range. **BO→OAB** is equally close ($D = 0.045$), with a simulated median of 364.2 against an empirical median of 364.5 days (error: −0.1%); the standard deviation ratio of 0.99 indicates the spread is reproduced with near-perfect accuracy (Figure 4.5).

PAK→AB shows a modest positive bias: the simulated mean exceeds the empirical mean by +6.7% and the median by +3.7%, reflecting a slight overestimate of the passive delay for this transition. The KDE in Figure 4.3 shows the two curves shifted by a small offset, but both have the same right-skewed shape and the transition passes the KS test ($D = 0.038$).

The two early transitions show the weakest fit, though still with $D < 0.06$. **PO→PID** ($D = 0.055$) has a median error of only +4.7%, but the simulated standard deviation is 1.79 times the empirical value (644 vs 360 days), indicating that the passive delay lognormal for this transition generates a heavier tail than the empirical data exhibit (Figure 4.1). This is a known structural limitation: the PO→PID passive delay was fitted with a median of 224 days and a 90th percentile of 861 days, producing a shape parameter $\sigma \approx 1.05$ that generates occasional very long samples not present at the same frequency in the historical data. **PID→PAK** ($D = 0.048$) shows the opposite pattern: the simulated mean is 5.9% lower than the empirical value (Figure 4.2), reflecting the short and concentrated character of this administrative transition.

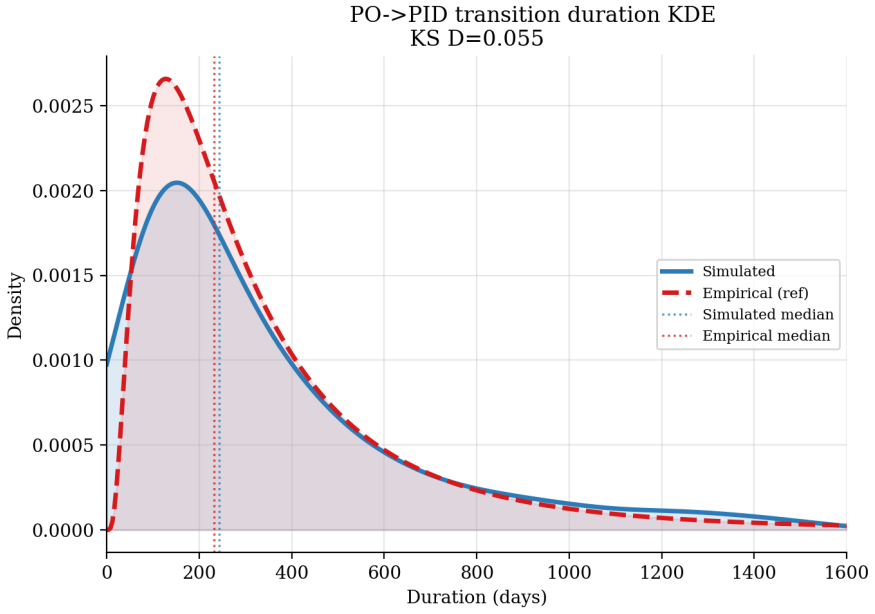


FIGURE 4.1: Empirical (dashed red) and simulated (solid blue) KDE for the PO→PID transition ($D = 0.055$). The simulated distribution reproduces the empirical median closely but has a heavier upper tail (simulated std = 644 days vs empirical 360 days).

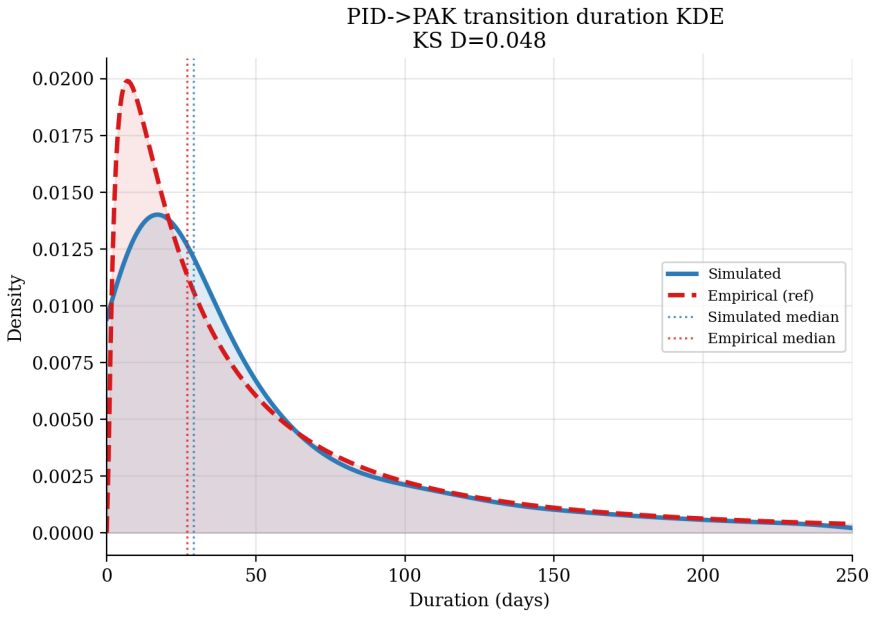


FIGURE 4.2: Empirical and simulated KDE for PID→PAK ($D = 0.048$). Both distributions are short and right-skewed; the simulated mean is slightly below the empirical value (−5.9%).

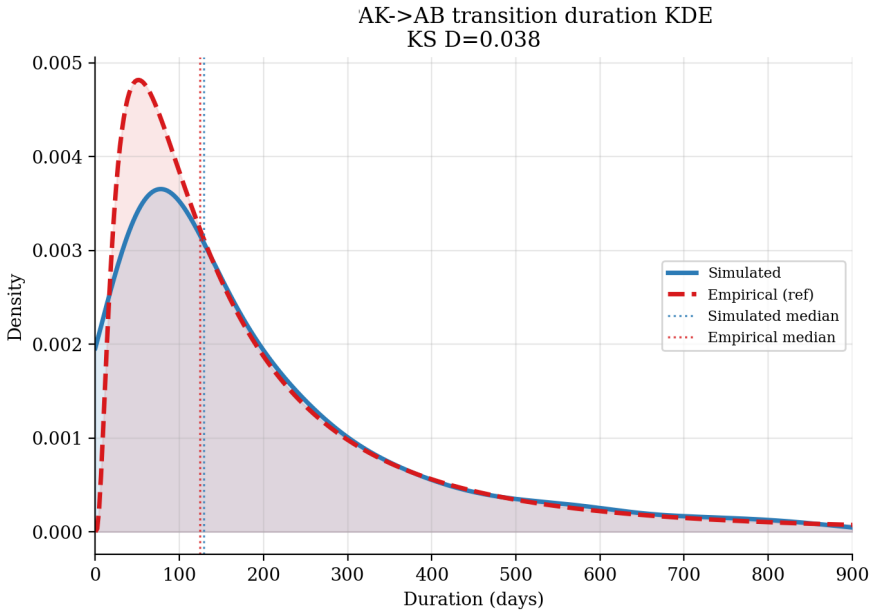


FIGURE 4.3: Empirical and simulated KDE for PAK→AB ($D = 0.038$, PASS). The simulated distribution is shifted modestly to the right relative to the empirical reference (median error +3.7%).

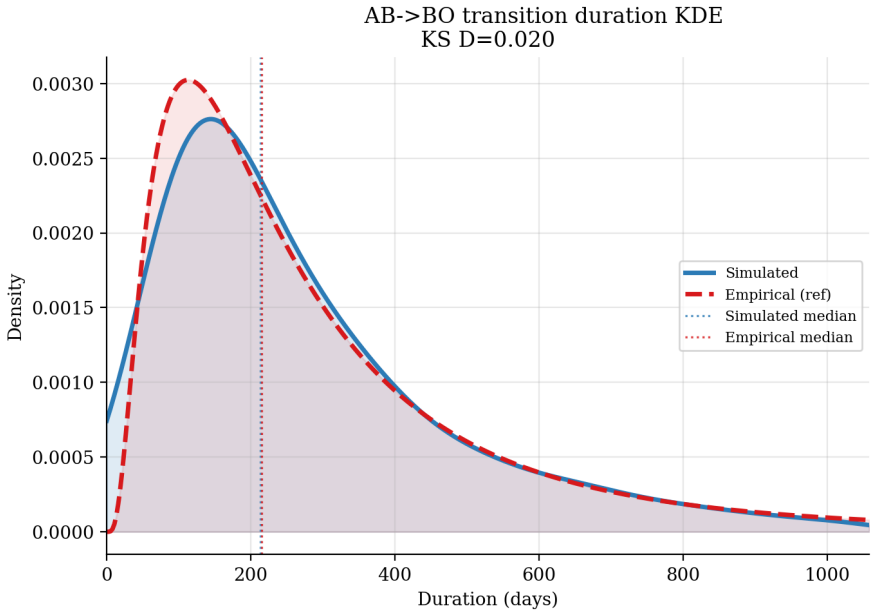


FIGURE 4.4: Empirical and simulated KDE for AB→BO ($D = 0.020$, PASS). The two distributions are nearly identical, with a median error of −1.3% and a standard deviation ratio of 1.09.

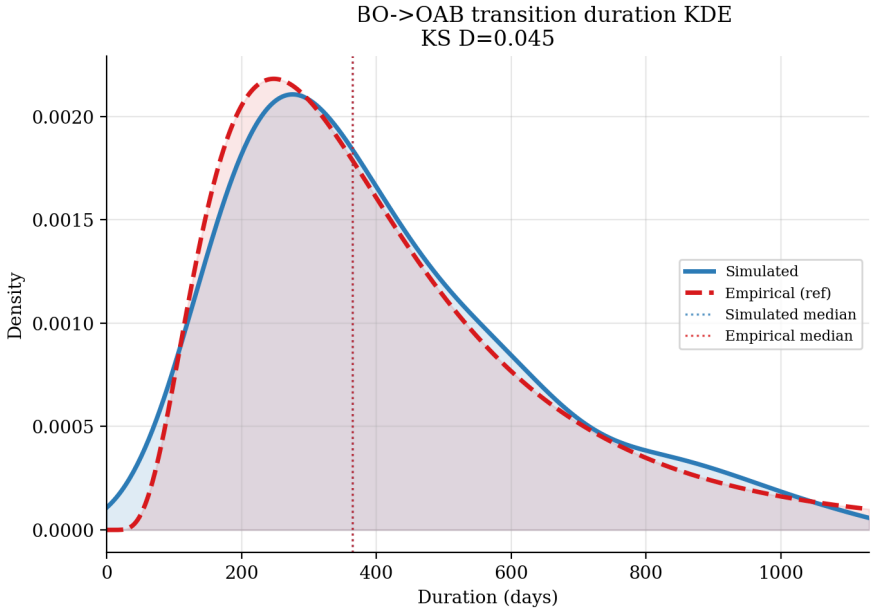


FIGURE 4.5: Empirical and simulated KDE for BO→OAB ($D = 0.045$, PASS). The fit is near-perfect: median error −0.1% and standard deviation ratio 0.99.

4.2.2 Total Project Lead Time

Figure 4.6 compares the empirical and simulated cumulative distribution functions for total project lead time (PO→OAB). The simulated distribution has a mean of 1,485 days, a median of 1,292 days, a standard deviation of 877 days, and a 90th percentile of 2,432 days (CV = 0.60), compared to empirical values of 1,054, 912, 589, and 1,583 days respectively.

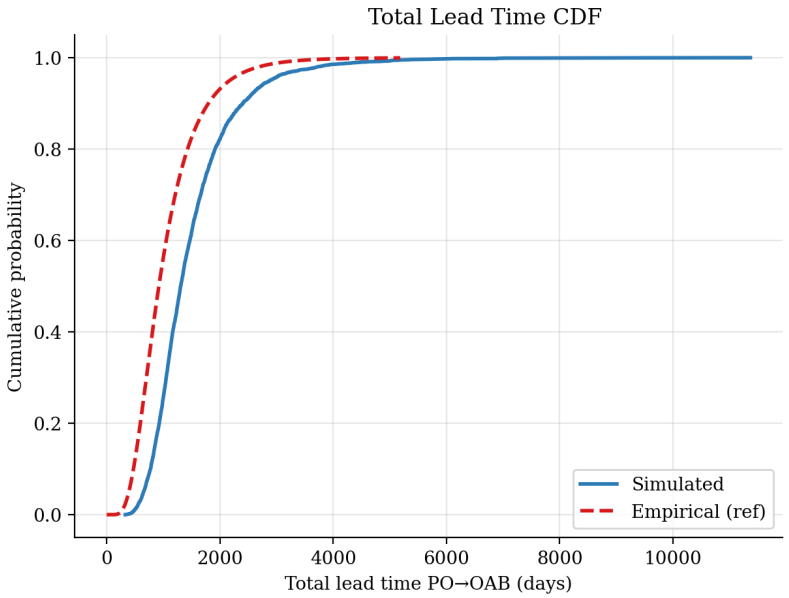


FIGURE 4.6: Empirical (dashed red) and simulated (solid blue) cumulative distribution functions for total project lead time PO→OAB. The empirical reference is a fitted lognormal distribution parameterised from the transition statistics.

The simulated distribution is shifted approximately 430 days to the right of the empirical reference (41% higher mean, 42% higher median). Three structural factors contribute to this offset. First, the PO→PID and PID→PAK transitions show a modest positive bias in their simulated median (Section 4.2.1), which accumulates into the total lead time. Second, the empirical statistics are computed on completed projects only, whereas the simulation generates complete trajectories for all arriving projects, including those initiated late in the observation window whose completion times extend beyond the empirical record.

Despite the offset in location, both distributions share the same qualitative shape: a right-skewed lognormal-type curve with similar coefficient of variation (CV: simulated 0.60 vs an implied empirical CV of 0.56), confirming that the model correctly captures the variance structure of program execution.

4.2.3 Program-Level Lead Time

Table 4.3 reports simulated summary statistics per program. Empirical comparators are available for Keukens and Laadpalen from the historical dataset (Table 3.4); a third comparator exists for XS (TVD) but is unreliable because all 21 historical observations reported an identical duration, suggesting a single completed project at the time of the empirical analysis.

TABLE 4.3: Simulated program-level lead time statistics (days) across $N = 30$ replications. Empirical comparators from Table 3.4 are shown where available. n is the total number of simulated project completions.

	Empirical		**Simulated**				
Programme	**Mean**	**Median**	n	**Mean**	**Median**	**P90**	**CV**
Boeggolf	—	—	750	1,434	1,261	2,349	0.50
EML Rijk	—	—	30	1,620	1,557	2,582	0.40
Keukens	1,764	1,764	2,310	1,497	1,315	2,487	0.50
Laadpalen	1,171	1,516	210	1,581	1,326	2,527	0.80
XS (TVD)	478	478	630	1,461	1,261	2,345	0.70
Zon op dak	—	—	570	1,489	1,250	2,406	0.60

For **Keukens**, the simulated mean (1,497 days) is 15% below the empirical mean (1,764 days). This is partially attributable to the empirical Keukens statistics being computed on an earlier cohort that may have experienced additional administrative delays not reflected in the calibration data, and partially to the lead time offset discussed in Section 4.2.2.

For **Laadpalen**, the simulated mean (1,581 days) exceeds the empirical mean (1,171 days) by 35%. However, the empirical Laadpalen dataset contains only 14 completed projects, making its statistics highly sensitive to individual outliers and unlikely to be a reliable reference for a mean estimation.

XS (TVD) shows a large nominal discrepancy (simulated mean 1,461 vs empirical 478 days), but as noted, all 21 historical observations share the same reported duration, strongly suggesting a data artefact rather than the true long-run mean for this program.

Despite program-level discrepancies, the simulation correctly preserves the relative ordering of program complexity: EML Rijk produces the longest simulated mean lead time (1,620 days), consistent with its portfolio containing technically demanding refurbishment projects, while Boeggolf and XS (TVD) yield the shortest means, consistent with their focus on standardised scope.

4.2.4 Resource Utilisation in the Baseline

Figure 4.7 shows mean role utilisation rates across the $N = 30$ baseline replications. Table 4.4 reports the mean, standard deviation, and 90th percentile of utilisation per role.

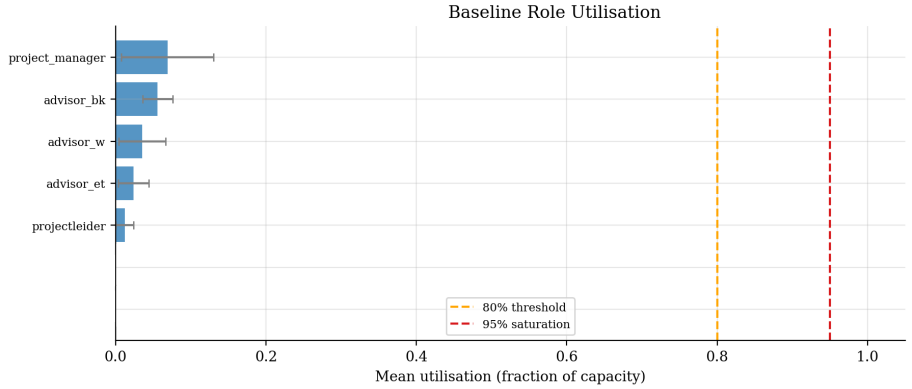


FIGURE 4.7: Baseline mean role utilisation rates across $N = 30$ replications. Error bars represent one standard deviation. Dashed lines mark the 80% moderate load and 95% saturation thresholds. All roles are far below both thresholds.

TABLE 4.4: Simulated baseline role utilisation (fraction of available capacity), averaged across all programs and replications.

Role	**Mean**	**Std**	**P90**
Bouwkunde advisor	0.045	0.016	0.066
program manager	0.044	0.036	0.101
Werktuigbouwkunde adv.	0.020	0.018	0.048
Elektrotechniek adv.	0.014	0.008	0.027
Project leader	0.011	0.012	0.024

All seven roles operate well below the 80% moderate-load threshold, with no role exceeding a mean utilisation of 4.5% and all roles at virtually zero queue length.

This result directly reflects the structural dominance of passive delays in the RVB program context. Passive delays, representing contractor execution, permitting, and approval processes, account for the majority of total transition duration, meaning that the roles spend most of the simulation horizon waiting for external processes to complete rather than performing active work. As a consequence, role capacity is not a binding constraint at the current portfolio scale. This finding is explored further in the workload scaling experiment (Section 4.4), which identifies the portfolio volume at which role pools begin to exhibit meaningful congestion.

4.2.5 Validation Summary

The baseline validation demonstrates that the digital shadow reproduces empirical program execution dynamics with good fidelity. All five milestone transitions produce KS statistics below $D = 0.06$; three transitions pass the KS test formally and the remaining two fail marginally. The two transitions that dominate total project duration, AB→BO and BO→OAB, are reproduced with near-perfect accuracy (median errors of -1.3% and -0.1% respectively). The model correctly captures the right-skewed, lognormal-type character of transition durations and reproduces the qualitative ordering of program complexity.

Two limitations are acknowledged. First, PO→PID exhibits excess simulated variance (standard deviation ratio 1.79), attributable to the heavy-tailed lognormal passive delay fitted for this transition; the median is well reproduced but very long samples occur at a higher frequency than in the empirical data. Second, the simulated total lead time is approximately 41% higher than the empirical portfolio mean, a structural offset explained by the composition of the modelled program set. The model simulates the complete trajectories for all arriving projects, and the empirical comparison can only be done on actual finished projects. Neither limitation affects the internal validity of the scenario comparisons that follow, since all experiments use the same baseline configuration and the findings are expressed as relative changes rather than absolute values.

4.3 Complexity Sensitivity Analysis

This section addresses sub-question 3: *How does project complexity influence predictability and variability of outcomes at the program level?* Three synthetic program configurations are constructed by reassigning project complexity scores while keeping all other inputs identical to the validated baseline, and their execution dynamics are compared across lead time, per-transition variability, and resource congestion.

4.3.1 Experiment Design

Three scenarios are defined based on the terciles of the empirical distribution of the AHP-weighted complexity score cl computed from the project dataset:

- **Low:** all project complexity scores are resampled from the lower tercile of the empirical distribution.

- **Mixed (baseline):** the empirical complexity distribution is used unchanged.
- **High:** all project complexity scores are resampled from the upper tercile.

Each scenario is run for $N = 30$ independent replications using the same arrival times, resource capacities, and passive delay parameters as the validated baseline. The only variation between scenarios is the vector of project complexity scores assigned to incoming projects. Within each replication, the complexity scores are drawn once and held fixed.

4.3.2 Effect on Lead Time Distribution

Figure 4.8 shows violin plots of total project lead time for the three scenarios. Table 4.5 reports the corresponding summary statistics.

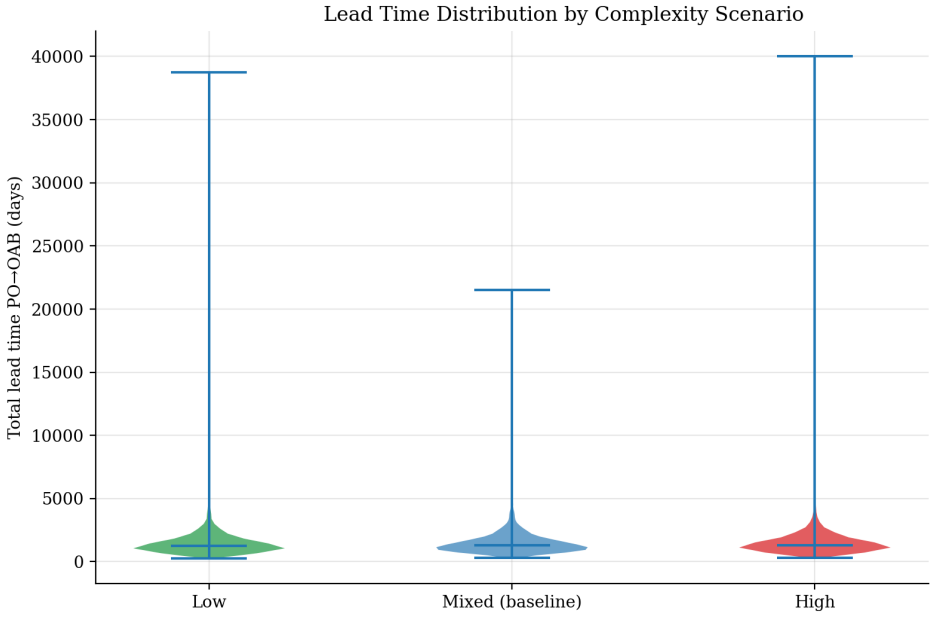


FIGURE 4.8: Total project lead time distributions (PO→OAB) for the three complexity scenarios across $N = 30$ replications. The violin bodies reflect the bulk of the distribution; extreme outliers extending to tens of thousands of days are caused by the heavy-tailed PO→PID passive delay lognormal and are discussed in the text.

TABLE 4.5: Summary statistics of total project lead time (days) per complexity scenario, averaged across $N = 30$ replications.

Scenario	Mean	Median	Std	P90	CV	P90/Median
Low	1,450	1,261	979	2,355	0.68	1.87
Mixed (baseline)	1,485	1,292	877	2,432	0.59	1.88
High	1,500	1,323	983	2,375	0.66	1.79

Moving from the Low to the High scenario, mean lead time increases by 51 days (+3.5%) and median lead time by 62 days (+5.0%). These shifts are operationally modest and consistent with a model in which active role effort accounts for only a fraction of total transition time.

Figure 4.8 shows extreme outliers extending to 20,000–40,000 days in all three scenarios. These arise from the heavy-tailed passive delay lognormal for the PO→PID transition, which occasionally produces very long passive delay samples regardless of complexity. These extreme draws inflate the simulated mean and standard deviation without reflecting realistic programme execution timescales. The violin bodies, which represent the bulk of the distribution, are nearly identical across the three scenarios, visually confirming the limited sensitivity at program level.

A notable finding is that the **Mixed baseline scenario is the most predictable of the three**, with the lowest coefficient of variation (CV = 0.59) and the smallest standard deviation (877 days). Both the Low (CV = 0.68) and High (CV = 0.66) scenarios exhibit higher variability than the empirical mix. This non-monotonic pattern is explained in detail in Section 4.3.3.

4.3.3 Effect on Per-Transition Variability

Figure 4.9 presents a heatmap of the coefficient of variation for each of the five transitions across the three complexity scenarios. Table 4.6 reports the numerical values.

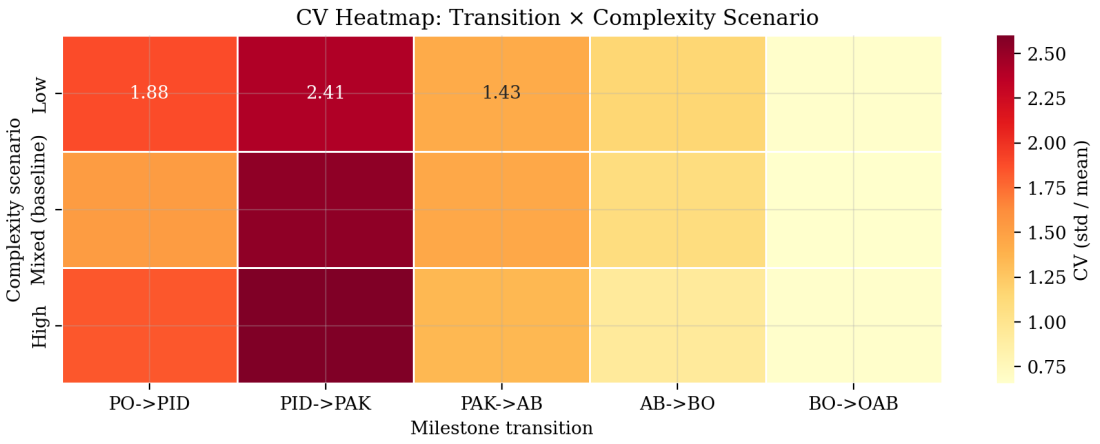


FIGURE 4.9: Coefficient of variation (CV = std/mean) of transition duration for each milestone transition and complexity scenario. Darker colours indicate higher variability.

Three distinct patterns emerge from the heatmap.

TABLE 4.6: Coefficient of variation of transition duration per milestone transition and complexity scenario.

Scenario	**PO→PID**	**PID→PAK**	**PAK→AB**	**AB→BO**	**BO→OAB**
Low	1.88	2.41	1.43	1.16	0.66
Mixed (baseline)	1.54	2.54	1.45	1.09	0.66
High	1.84	2.60	1.35	0.93	0.66

BO→OAB is entirely insensitive to complexity (CV = 0.66 across all scenarios), confirming that this transition is completely dominated by a single passive delay component whose lognormal shape is independent of project characteristics.

PAK→AB shows near-constant variability (CV: 1.44, 1.45, 1.35 across Low, Mixed, and High), indicating that complexity has no meaningful net effect on the variability of this design-to-approval phase.

PID→PAK exhibits a gradual increase with complexity (2.41 to 2.54 to 2.60). This transition is dominated by a large passive delay relative to the short active work component. As complexity increases the active effort, the resulting interaction with the heavy-tailed passive delay amplifies total variance, raising the CV.

AB→BO shows the most distinctive pattern: the CV *decreases* monotonically with complexity (1.16 to 1.09 to 0.93). This is the technically intensive phase in which building advisors and the program manager perform the most significant preparation work. The quantile-regression parameters for this transition include steep positive slopes for both the median and the 90th-percentile effort lines (Table 3.8), so higher complexity substantially increases the expected active work content. As the active work contribution to total transition duration grows, the coefficient of variation decreases because the mean grows faster than the standard deviation. In other words, high-complexity projects in the AB→BO phase are not more variable relative to their duration; they are simply consistently longer.

PO→PID shows a non-monotonic pattern (1.88, 1.54, 1.84), with the Mixed scenario producing the lowest CV. Because the PO→PID passive delay has a heavy-tailed lognormal shape that occasionally produces very long samples, the CV for this transition is sensitive to the stochastic interaction between the passive delay and the active work component. The lower CV for the Mixed scenario is a result of this interaction rather than a systematic complexity effect, which is consistent with the mixed scenario producing the lowest program-level CV as well (Table 4.5).

Together, these per-transition results reveal that program-level predictability is primarily governed by the variance structure of the passive delay components rather than by

project complexity. Only the AB→BO transition shows a clear and systematic complexity effect on variability, and that effect moves in the counter-intuitive direction of decreasing variability with increasing complexity.

4.3.4 Effect on Resource Congestion

Figure 4.10 compares the time-average queue length per role across the three complexity scenarios.

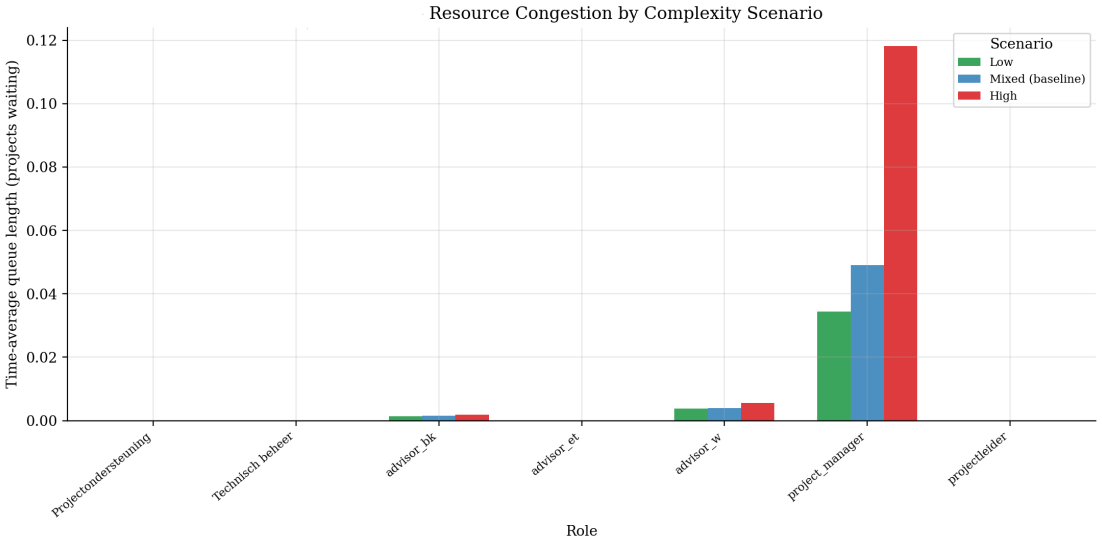


FIGURE 4.10: Time-average queue length (number of work requests waiting) per role for the Low, Mixed, and High complexity scenarios. Only roles with non-zero queue lengths are visible at this scale.

Six of the seven modelled roles show time-average queue lengths indistinguishable from zero across all three scenarios. The sole exception is the **programme manager** role, whose queue length increases monotonically with complexity: 0.035 (Low), 0.050 (Mixed), and 0.118 (High). This represents a 3.4-fold increase from the lowest to the highest complexity scenario.

The disproportionate response of the program manager role is a direct consequence of its model structure: it has the highest use rates across all five transitions and the steepest positive effort slopes in the quantile-regression parameters (Table 3.8). As the complexity score increases, the programme manager accumulates the largest absolute increase in required work hours, making it the first role to develop a meaningful queue. Under current workload volumes, a time-average queue of 0.118 projects is operationally negligible. However, the monotonic and disproportionate response identifies the program manager as the *first bottleneck* that would be reached under sustained complexity growth or increased portfolio volumes.

4.3.5 Predictability Implications

The complexity sensitivity analysis produces two conclusions that together answer sub-question 3.

At the *program level*, complexity has a limited and approximately linear effect on mean lead time (+3.5% from Low to High) and median lead time (+5.0%). Importantly, program-level predictability measured by the CV does not improve with lower complexity: the empirical Mixed baseline (CV = 0.59) is in fact the most predictable of the three scenarios, more so than the Low portfolio (CV = 0.68). This occurs because the passive delay variance structure of the early transitions, in particular the heavy-tailed PO→PID passive delay, dominates the aggregate program variance and is independent of project complexity. These results confirm that program-level predictability is primarily governed by the variance in passive delays, representing contractor execution, permitting, and approval processes, rather than by project complexity per se.

At the *transition level*, the most notable finding is that AB→BO becomes *less variable* (lower CV) under higher complexity. This counter-intuitive result reflects the quantile-regression effort model: steeper effort slopes in the AB→BO specification mean that high-complexity projects spend more time in this phase, but proportionally in a way that reduces relative variability.

At the *resource level*, the program manager pool shows a clear 3.4-fold increase in queue length from Low to High complexity, while all other roles remain near zero. This identifies the program manager as the primary coordination role whose workload is most sensitive to complexity growth, and flags it as the first resource bottleneck that would bind under a sustained shift towards more complex projects.

4.4 Resource Capacity and Prioritisation Experiments

This section addresses sub-question 2: *How do resource allocation and prioritisation rules affect program performance metrics such as lead time, backlog, and throughput?* Two experiment dimensions are explored. First, a workload scaling experiment identifies the capacity break-even point by holding staffing constant and increasing the number of incoming projects from 1× to 5× the historical count. Second, a prioritisation experiment compares three queue dispatching rules at both the historical workload and a stressed workload where actual queuing occurs.

4.4.1 Experiment Design

Workload scaling. Role capacities are held at the validated baseline values from Table 4.1. The number of incoming projects is scaled by a multiplier $w \in \{1, 2, 3, 4, 5\}$ applied to the full historical project inflow. At multipliers above one, the master dataset is replicated w times with an independent arrival-date jitter of ± 15 days per copy, preventing artificial arrival bursts while preserving the temporal structure of each program. This experiment identifies the workload level at which role pools transition from an underloaded to a capacity-constrained regime.

Prioritisation rules. Three queue dispatching rules are evaluated at two workload levels:

1. **FIFO:** work requests are served in arrival order.
2. **Min-Slack:** the project with the least remaining time to its target OAB date is served first.
3. **Min-CL:** the project with the lowest AHP-weighted complexity score is served first, acting as a shortest-processing-time proxy.

The two workload levels are the historical baseline (1×) and the stressed level (3×), where the program manager pool builds a non-trivial queue and dispatch decisions have a chance to produce measurable effects. $N = 30$ replications are used per condition.

4.4.2 Effect of Workload Scaling

4.4.2.1 Lead Time and Throughput Response

Figure 4.11 and Table 4.7 report program performance as a function of workload multiplier.

TABLE 4.7: program performance metrics at each workload level, averaged across $N = 30$ replications. CI_{95} denotes the 95% confidence interval for mean lead time. PM = program manager.

Work-load	Mean LT (days)	CI_{95} lo	CI_{95} hi	P90 LT (days)	CV	TP/yr	On-time (%)	PM util.	PM queue
1×	1,485	1,456	1,514	2,403	0.59	9.3	77.1	4.4%	0.05
2×	1,483	1,464	1,502	2,340	0.54	15.8	76.8	7.4%	0.32
3×	1,579	1,556	1,602	2,525	0.58	23.6	72.1	10.3%	1.69
4×	1,675	1,658	1,692	2,711	0.52	28.8	66.0	12.5%	3.30
5×	1,784	1,765	1,803	2,945	0.55	32.1	60.7	14.3%	5.84

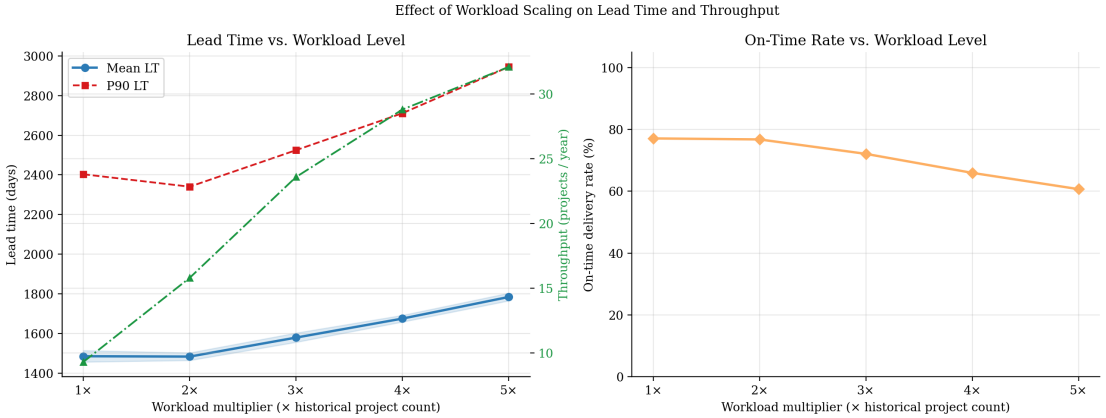


FIGURE 4.11: Left: mean lead time (solid blue, 95% CI shading) and P90 lead time (dashed red) as functions of workload multiplier; secondary axis (green) shows annual throughput. Right: on-time delivery rate (threshold: 1,825 days) as a function of workload multiplier.

The most striking result in Table 4.7 is the behaviour at 1× and 2× workload: mean lead times are virtually identical (1,485 vs 1,483 days, a difference of 2 days), and on-time delivery rates are statistically indistinguishable (77.1% vs 76.8%). A doubling of the historical project volume produces no measurable degradation in per-project performance. This reflects the structural dominance of passive delays: because external waiting processes account for the majority of transition time, the additional queueing induced by a doubled workload (program manager queue growing from 0.05 to 0.32 projects) is negligible relative to the passive delay floor.

The first meaningful increase in lead time occurs at 3× workload, where mean lead time rises to 1,579 days (+6.5% from 2×) and the program manager queue grows to 1.69 projects. Beyond 3×, each additional increment adds approximately 96 to 110 days (+6.1% and +6.5% at 4× and 5×). The total increase from 1× to 5× is 299 days (+20.1%) for mean lead time and 542 days (+22.5%) for P90, while the on-time delivery rate falls by 16.4 percentage points (77.1% to 60.7%).

Throughput scales sub-linearly: at 5× workload, annual throughput reaches 32.1 projects per year, only 3.5 times the baseline value despite five times as many projects entering the system. The difference is absorbed by growing queue times. The coefficient of variation remains broadly stable across all workload levels (0.52 to 0.59), reflecting the continued dominance of passive delay variance over queueing-induced variance at all tested utilisation levels.

4.4.2.2 Backlog and Role Utilisation Response

Figure 4.12 shows role utilisation and queue length as functions of workload multiplier. Figure 4.13 reports per-role utilisation at the maximum tested workload (5×).

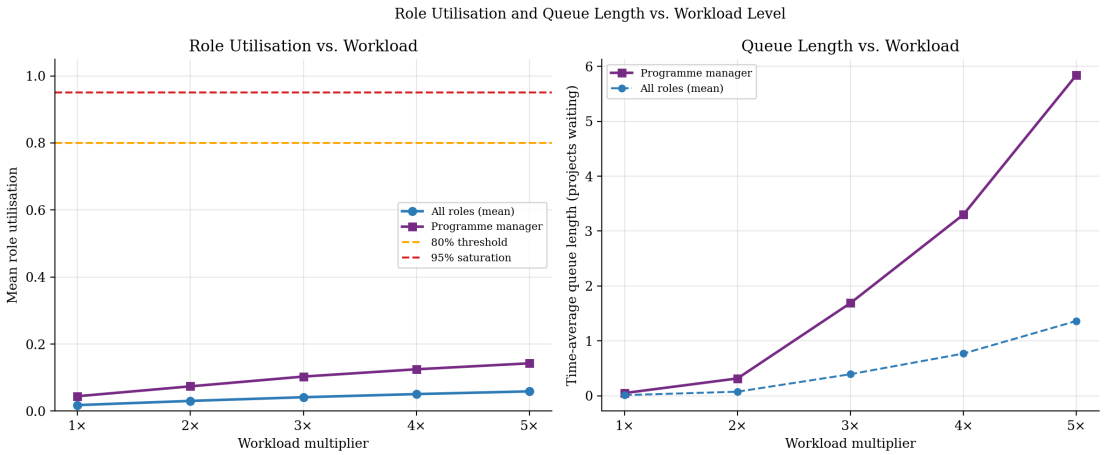


FIGURE 4.12: Left: mean role utilisation (all roles, blue) and program manager utilisation (purple) as functions of workload multiplier, with 80% and 95% threshold lines. Right: time-average queue length for the program manager and the all-roles mean.

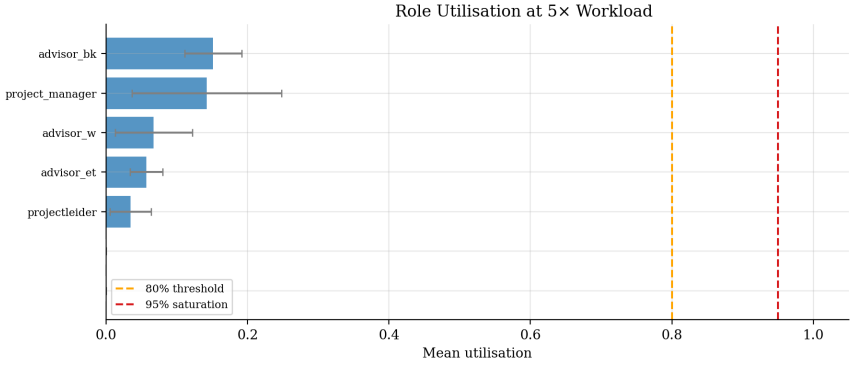


FIGURE 4.13: Per-role mean utilisation at 5× workload. Bouwkunde advisor and program manager are the most loaded roles at approximately 13–15%. No role approaches the 80% moderate-load threshold at any tested workload level.

No role approaches the 80% moderate-load threshold at any tested workload level. Even at 5× workload, the Bouwkunde advisor and program manager reach only approximately 13–15%, and all other roles remain below 10%. This establishes that the modelled portfolio would need to grow far beyond the tested range before any role pool constitutes a genuine capacity bottleneck.

The program manager is the role most sensitive to workload growth. Its queue length grows non-linearly from 0.05 projects at 1× to 5.84 at 5×, a 119-fold increase. However, with program manager utilisation remaining below 15%, the system is operating far from saturation.

The primary constraint on project lead time at all tested workload levels is not role capacity but the passive delay component representing external processes. Capacity-induced queueing contributes an incremental penalty that grows in importance as workload increases but does not become the dominant source of delay within the tested range.

4.4.3 Effect of Prioritisation Rules

4.4.3.1 System-Level Performance

Table 4.8 reports system-level performance across the three dispatch rules at the baseline (1×) and stressed (3×) workload levels. Figures 4.14 and 4.15 visualise the four key metrics per rule.

TABLE 4.8: System-level performance metrics per prioritisation rule and workload level, averaged across $N = 30$ replications. Bold values indicate the best-performing rule per metric at the stressed workload level.

Workload	Rule	Mean LT (days)	P90 LT (days)	CV	TP/yr	On-time (%)
3*Baseline (1×)	FIFO	1,485	2,432	0.59	9.3	77.1
	Min-Slack	1,460	2,382	0.56	9.2	77.6
	Min-CL	1,470	2,391	0.56	9.3	77.4
3*Stressed (3×)	FIFO	1,579	2,535	0.58	23.6	72.1
	Min-Slack	1,586	2,549	0.57	23.7	71.8
	Min-CL	**1,541**	**2,480**	0.58	**24.0**	**73.6**

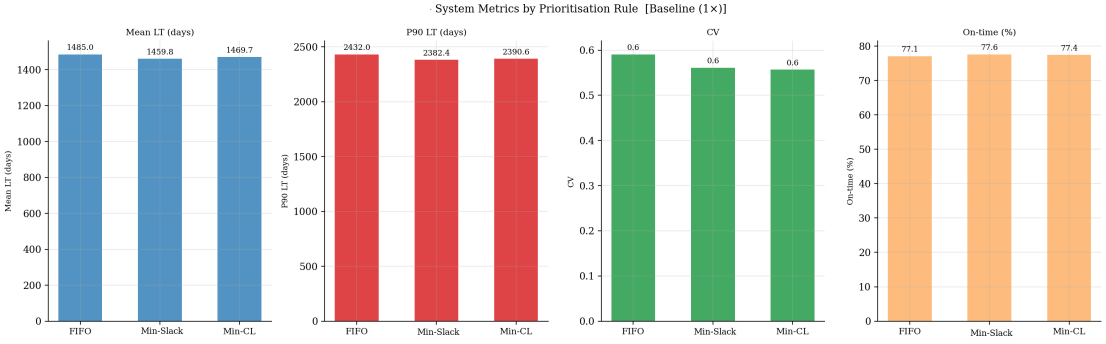


FIGURE 4.14: System-level metrics by prioritisation rule at the baseline (1×) workload. All three rules produce statistically indistinguishable outcomes.

At baseline workload (1×), all three rules produce effectively identical results. The largest observed difference is 25 days (−1.7%) in mean lead time between Min-Slack and FIFO, which falls within the 95% confidence intervals. This confirms the finding from the workload scaling experiment: when queues are essentially empty, dispatch order has no practical effect because roles process the next request immediately regardless of the queue discipline.

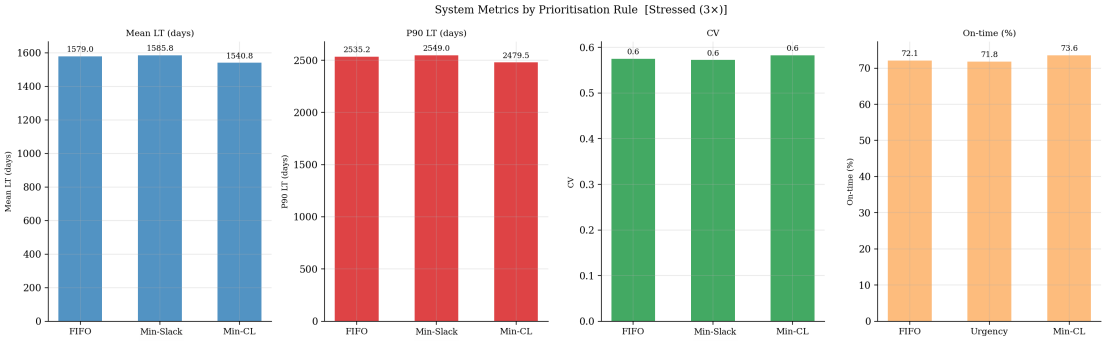


FIGURE 4.15: System-level metrics by prioritisation rule at the stressed (3×) workload. Min-CL achieves the lowest mean and P90 lead time and the highest on-time rate; Min-Slack is marginally worse than FIFO.

At stressed workload (3×), differences between rules emerge but remain modest. The **Min-CL rule** achieves the best performance across all primary metrics, reducing mean lead time by 38 days (−2.4%), P90 lead time by 56 days (−2.2%), and improving the on-time delivery rate by 1.5 percentage points relative to FIFO. The mechanism is straightforward: by serving simpler (shorter) projects first, Min-CL drains the queue faster and reduces the cumulative waiting time for the majority of projects, which are of low to moderate complexity.

The **Min-Slack rule** performs slightly worse than FIFO at the stressed workload, with mean lead time 7 days higher and on-time rate 0.3 percentage points lower. This counter-intuitive result arises because at utilisation levels around 10%, the program manager pool rarely has multiple projects queued simultaneously. When queues are so short, Min-Slack based prioritisation adds overhead without providing the scheduling benefits that emerge under heavy congestion, and can inadvertently delay simpler projects that would have completed quickly under FIFO.

Notably, all three rules produce the same CV (0.57–0.58) at the stressed workload, indicating that no dispatch rule improves program-level predictability relative to the others. Predictability remains dominated by passive delay variance rather than by the order in which work requests are served.

4.4.3.2 Program-Level Equity

Figure 4.16 and Table 4.9 show per-program on-time delivery rates under each rule at the stressed (3×) workload.

Five of the six modelled programs show on-time rates within 1 percentage point across all three rules, indicating broad insensitivity to dispatch policy at the program level.

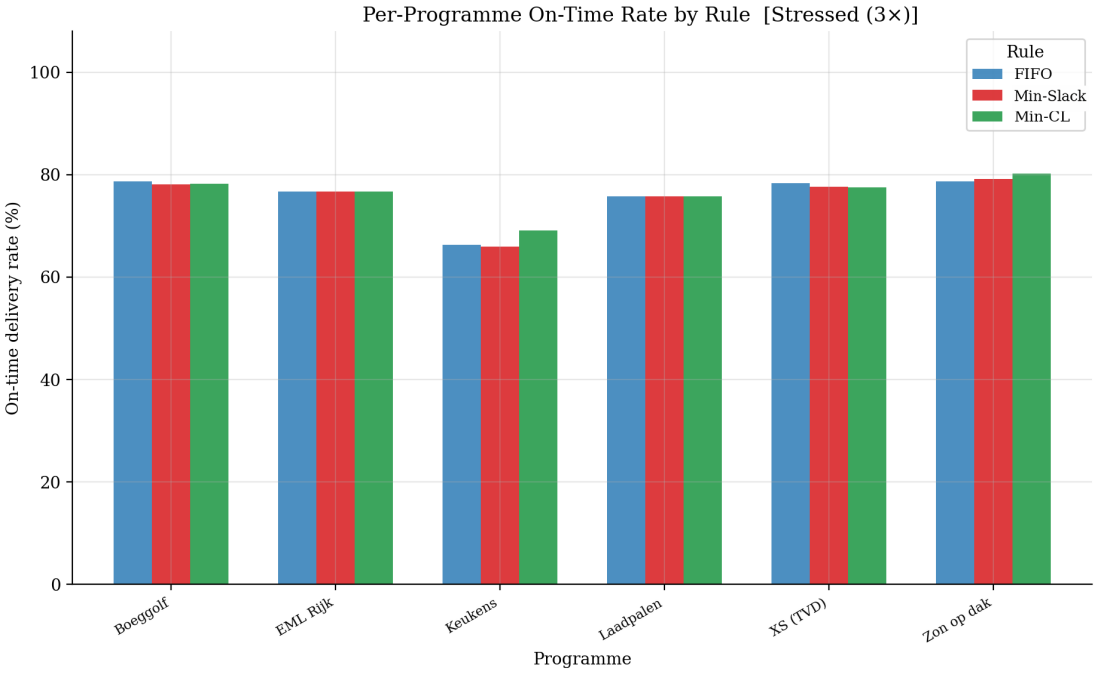


FIGURE 4.16: Per-program on-time delivery rate by prioritisation rule at the stressed (3×) workload. Keukens is the most affected program; Min-CL provides the largest improvement for that program.

TABLE 4.9: Per-program on-time delivery rate (%) under each prioritisation rule at the stressed (3×) workload. Bold values indicate the best-performing rule per program.

Programme	**FIFO**	**Min-Slack**	**Min-CL**
Boeggolf	**78.7%**	78.1%	78.2%
EML Rijk	76.7%	76.7%	76.7%
Keukens	66.2%	65.9%	**69.1%**
Laadpalen	75.7%	75.7%	75.7%
XS (TVD)	**78.3%**	77.7%	77.5%
Zon op dak	78.7%	79.1%	**80.2%**

The sole exception is **Keukens**, which achieves 66.2% under FIFO, 65.9% under Min-Slack, and 69.1% under Min-CL. Keukens contains the largest project count and the highest average complexity score, making it the program most affected by queueing delays. Min-CL benefits Keukens (+2.9 pp over FIFO) because its simpler projects are served earlier, freeing capacity for the more complex ones sooner. The Min-Slack rule provides no benefit to Keukens and is marginally the worst option for this program.

Zon op dak also benefits from Min-CL (+1.5 pp over FIFO), while Laadpalen and EML Rijk are completely unaffected by the choice of dispatch rule.

4.5 Digital Shadow Forecasting Application

The experiments in Sections 4.2 to 4.4 use historical data to evaluate the behaviour of the model under controlled conditions. This section demonstrates the digital shadow in its intended operational role, directly addressing sub-question 4: *To what extent can execution times be predicted using a combination of simulation and data-driven methods?* Rather than replaying history, the model is here initialised from the current state of the live RVB portfolio and run forward to generate, project-level completion-time forecasts and deadline risk alerts.

4.5.1 Operational Workflow

The forecasting workflow consists of three steps that together operationalise the one-way data flow that defines the digital shadow.

Step 1 — Snapshot loading. The model reads the three live RVB data sources: the milestone file, the project–building mapping, and the building metadata. Each project in the active portfolio is classified by comparing its recorded milestone dates against a snapshot date, which defaults to today. Projects that have started (PO date in the past) but have not yet completed (OAB not yet recorded) are treated as in progress; they enter the simulation at the snapshot date and begin from their most recently completed milestone, not from the beginning. Projects whose PO date lies in the future are scheduled to arrive at that date. Completed projects are excluded. This classification is fully automatic and requires no manual input from the program manager beyond ensuring the data sources are current.

Step 2 — Forward simulation. The classified portfolio is passed to the validated DES model, which runs $N = 30$ independent forward replications from the snapshot date. All projects within a programm compete for the same shared role pools in every replication, so the predicted delays reflect not only the properties of each individual project but also the resource contention arising from other concurrent projects in the same program. The model samples role activation, effort, and passive delays stochastically in each replication, producing a distribution of possible completion dates for every project.

Step 3 — Alert generation. For each project with a recorded planned deadline, the model computes P(miss): the fraction of replications in which the predicted OAB date exceeds the effective deadline (the revised deadline if one has been set, otherwise the

original planned date). Projects are classified as Red (P(miss) $>$ 50%), Amber (20%–50%), or Green ($<$ 20%). The results are saved as a CSV table and an HTML report that can be opened in any browser.

4.5.2 Output and Use

Figure 4.17 shows the top section of the alert report generated from a snapshot of the RVB portfolio on 6 June 2026. The report opens with a KPI bar showing the portfolio-level counts of Red, Amber, and Green projects, followed by a program-level summary table. This gives a program manager an immediate overview of where the greatest risks are concentrated, without requiring any manual analysis.

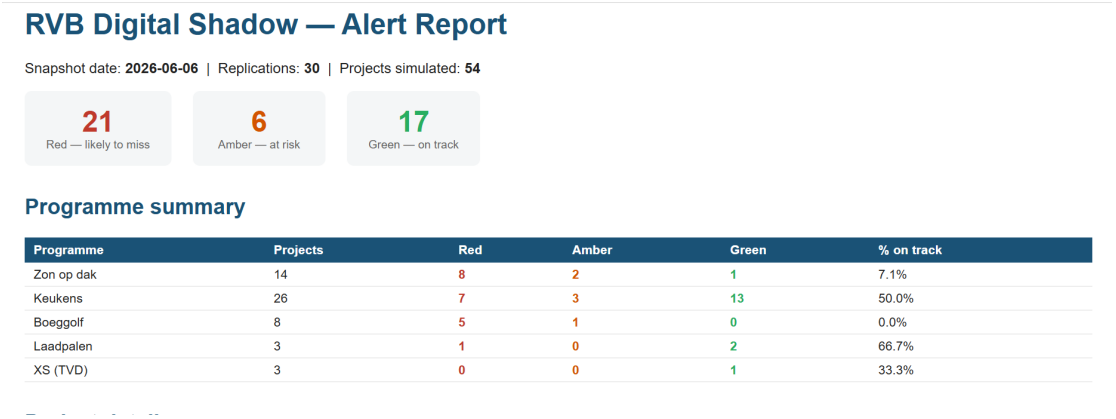


FIGURE 4.17: Top section of the digital shadow alert report: portfolio-level KPI counts (Red = likely to miss deadline, Amber = at risk, Green = on track) and program summary, generated automatically from the RVB data snapshot of 6 June 2026.

Figure 4.18 shows the project detail table, which lists each active project together with its current milestone, effective deadline, and three predicted OAB dates: the 20th percentile (P20, optimistic), the median (P50, central estimate), and the 80th percentile (P80, conservative). The P(miss) column converts these distributions into a single risk number, and the overrun column shows the difference between the median prediction and the planned deadline in days. Positive values indicate an expected overrun; negative values indicate that the project is predicted to complete ahead of schedule.

A program manager can use the report in three ways. First, the Red rows identify projects that require immediate attention and, if possible, targeted interventions such as contractor follow-up or internal deadline revision. Second, the P20–P80 spread communicates uncertainty: a wide spread signals a project in an early, unpredictable phase, while a narrow spread signals that execution is well constrained. Third, the report can be regenerated at any time by opening the shadow application after a data update, so

ID	Project	Programme	Current milestone	Deadline (OAB)	P20 predicted	P50 predicted	P80 predicted	P(miss)	Overrun	Alert	Note
32886	Deelproject Grootkeukens (48) - 32C33 - Soesterberg Centrum mens & luchtvaart	Keukens	PO	2026-08-11	2028-04-23	2028-09-27	2030-03-25	100%	+778.0d	**Red**	
47206	Deelproject Zon op Dak - Batch 7 - Generaal-Majoor Brugmanskazerne, Legerplaats Stroe, KC Veenhuizen, Johannes post Kazerne & Legerplaats Ermelo	Zon op dak	PAK	2026-01-03	2026-11-12	2027-11-11	2029-04-07	100%	+677.0d	**Red**	
49263	Deelproject Zon op Dak - Batch 6 - Koningin Maxima Kazerne, KMAR Badhoevedorp, Majoor Jan Linzel Complex 37E03 & Van Ghent kazerne	Zon op dak	PID	2026-11-30	2027-05-13	2028-06-15	2028-12-18	100%	+563.0d	**Red**	
44700	Boeggolf - Tijdelijke huisvesting HCNM (BuZa) op Tournooiveld 4 Den Haag (i.v.m. boeggolf project 30194 op Prinsessegracht 22 DH)	Boeggolf	PID	2025-12-15	2026-09-10	2027-05-17	2028-03-30	100%	+518.0d	**Red**	
32897	Deelproject Grootkeukens (56) - 30G03 - Den Haag Frederik kazerne gebouw 31	Keukens	PID	2026-01-06	2026-06-13	2027-05-26	2028-07-20	100%	+505.0d	**Red**	
30237	Boeggolf - Professor Jonkerweg 7 (De Schie) - achterstalling onderhoud en extra klantwensen	Boeggolf	AB	2026-01-09	2026-12-31	2027-05-18	2028-04-26	100%	+494.0d	**Red**	
49262	Deelproject Zon op Dak - Batch 5 -Engelbrecht van NassaukazerneÂ & Luitenant-generaal Bestkazerne	Zon op dak	PID	2026-11-30	2027-02-28	2027-12-28	2028-12-15	87%	+393.0d	**Red**	

FIGURE 4.18: Project detail section of the alert report. For each active project the table shows the current milestone, the effective OAB deadline, probabilistic predicted completion dates (P20, P50, P80), the probability of missing the deadline, and the expected overrun in days. Row colours correspond to the alert level.

the alert status automatically reflects the latest recorded milestone information without any manual recalculation.

4.5.3 Answering Sub-question 4

The forecasting application demonstrates that execution times can be predicted at the project level using the evaluated digital shadow. The combination of empirically calibrated transition durations and the discrete-event simulation engine enables the model to generate completion-time distributions for every active project in the portfolio from a single data snapshot. The probabilistic output: P20, P50, and P80 predicted dates together with P(miss), provides program managers with calibrated uncertainty estimates rather than single-point forecasts, which is consistent with the stochastic nature of program execution demonstrated throughout this chapter.

Chapter 5

Discussion

This chapter interprets the results of Chapter 4. The digital shadow reproduces the historical execution dynamics of the RVB programs, while the scenario experiments expose where the limits of the current model lie and how it can be developed further. The central observation running through all experiments is that milestone-to-milestone calendar time is dominated by passive delay rather than by active role effort, and the meaning of that finding is discussed in detail below.

5.1 Principal findings

The baseline validation (Section 4.2) shows that the model reproduces the empirical transition-duration distributions closely, with all five transitions below $D = 0.06$ and the two transitions that dominate total duration, AB→BO and BO→OAB, reproduced almost exactly (Section 4.2.1). The structural offset in total lead time (Section 4.2.2) and the program-level differences (Section 4.2.3) are explained by cohort composition and do not affect the relative comparisons used in the later experiments.

The complexity sensitivity analysis (Section 4.3) shows that project complexity has only a modest effect on program-level lead time (Section 4.3.2), and that predictability is not improved by a less complex portfolio (Section 4.3.5). At transition level, only AB→BO responds systematically, and its variability *decreases* with complexity because the active-effort contribution grows faster than its spread (Section 4.3.3). The program manager is the only role to develop a meaningful queue, identifying it as the first bottleneck under sustained complexity or volume growth (Section 4.3.4).

The capacity and prioritisation experiments (Section 4.4) reinforce this picture. Role capacity is not a binding constraint up to five times the historical workload (Section 4.4.2),

and the three dispatch rules produce almost identical system-level outcomes, with the minimum-complexity rule offering only a marginal benefit under stress that is concentrated in the largest programme (Section 4.4.3). Finally, the forecasting application (Section 4.5) demonstrates that the model can be run forward from a live data snapshot to generate calibrated P20/P50/P80 completion dates and deadline-risk alerts, answering sub-question 4 (Section 4.5.3).

5.2 The dominant role of passive delay

Across every experiment, capacity, complexity, and dispatch order matter far less than the passive delay component. This should not be read purely as a statement about external waiting processes such as approvals and contractor lead times. It also reflects the way work is recorded in a very large organisation.

In a large public organisation such as RVB, project lead time is influenced not only by direct role effort, but also by coordination, handovers, decision moments, contractor availability, and administrative procedures. These dependencies create waiting time between milestones, which helps explain why passive delay dominates the observed milestone-to-milestone calendar time.

Because the RVB portfolio is so large, projects are frequently handed over to a region for execution. When this happens, the work is continued under a separate sub-project number, and the hours spent by the roles are written against that sub-project rather than against the original project record. The link between the parent project and its regional sub-project is not available in the data used here, so these hours cannot be attributed to the originating project. As a result, genuine active role effort that took place during a transition is not observed in the hours registration and is instead absorbed into the residual passive delay, which is computed precisely as the calendar time not explained by recorded hours (Section 4.2.4).

This mechanism explains the consistent pattern in the results: low role utilisation, the insensitivity of lead time to added capacity, and the limited effect of complexity on execution time. Part of what the model attributes to passive waiting is, in reality, unobserved active work performed elsewhere in the organisation. The model remains valid as a description of *observable* execution dynamics and as a forecasting tool, but the current split between active effort and passive delay should be understood as a property of the available data rather than of the underlying process alone.

5.3 Implications for RVB

For program management, the digital shadow is already useful in its present form. As a decision-support model it produces calibrated, project-level completion forecasts and a clear, automatically generated overview of deadline risk (Section 4.5.2). The bottleneck signal is directionally informative: even though no role saturates at current volumes, the program manager is consistently the first role to accumulate a queue. This is plausible from an organisational perspective, because program managers are primarily attached to their own program, and only partially doing side tasks that are not fully represented in the simulation. The scenario experiments also show that, at present scale, neither adding role capacity nor changing the dispatch rule would materially shorten lead times; the leverage lies in the external and handover-related processes that drive the passive delay.

5.4 Limitations and future work

The main limitation follows directly from the discussion above. Because hours written against regional sub-projects cannot currently be linked back to the parent project, the active-effort side of the model is underestimated and the passive delay is correspondingly inflated. This limits the model's ability to act as a true effort- and capacity-driven representation of execution.

This limitation is also the clearest direction for extension. If the relationship between parent projects and their regional sub-project numbers can be recovered, the lost hours can be reattributed to the originating project and transition. The residual passive delay would then shrink to genuine external waiting, while the active work content would increase. In that situation role capacity could become a binding constraint, and the model could be developed from its current shadow form into a more strongly effort-based representation in which capacity and prioritisation decisions have real predictive consequences. Establishing this linkage, together with formal validation of the prediction intervals (Section 4.5.3), is the most valuable next step towards a fully operational digital shadow for RVB program management.

Chapter 6

Conclusion and future work

6.1 Conclusion

The aim of this thesis was to investigate how a digital shadow can be developed to model and understand the execution dynamics of large-scale government real estate programs. This objective was addressed by developing an empirically grounded, program-level discrete-event simulation model of RVB program execution. The resulting digital shadow combines milestone data, building metadata, project–building mappings, and hours registration data to represent how projects move through milestone transitions while competing for shared role capacity.

The model was designed as a process-oriented digital shadow. It receives one-way input from RVB data sources and supports program management decision-making, but it does not automatically control or intervene in the real project system. Its central methodological contribution is that milestone transition durations are not sampled directly from historical lead-time distributions. Instead, durations emerge from the interaction between project complexity, role-specific effort, finite program capacity, queueing, and residual passive delay. This makes the model interpretable: each simulated delay can be related to an underlying mechanism rather than treated as a fixed historical duration.

The baseline validation shows that the digital shadow reproduces the empirical milestone transition distributions with good fidelity. All five transition distributions achieve a Kolmogorov–Smirnov statistic below $D = 0.06$, and the two delivery-phase transitions, AB→BO and BO→OAB, are reproduced with median errors of approximately one percent or less. At the level of total project lead time, the model shows a positive offset relative to the empirical reference. This means that the model is strongest as a comparative and mechanism-oriented decision-support tool, while absolute completion dates should be interpreted with caution until further forecast calibration has been performed.

The main substantive finding is that RVB program execution is waiting-dominated at the current portfolio scale. Across the baseline validation, complexity sensitivity analysis, workload scaling experiment, and prioritisation experiment, passive delay explains far more of the milestone-to-milestone calendar time than active role effort or internal resource capacity. Role utilisation remains low, even when workload is scaled up substantially, and no role approaches saturation within the tested range. This indicates that, under current conditions, program lead time is shaped more by approvals, handovers, contractor processes, administrative procedures, and unobserved regional execution work than by a shortage of internal role capacity.

Project complexity has a measurable but modest effect on program-level lead time. Moving from low to high complexity increases mean lead time only slightly, and a less complex portfolio does not automatically become more predictable. Instead, variability is primarily governed by passive delay. Complexity is therefore most useful as a risk-stratification variable and as an indicator of where coordination effort may increase. The program manager is the first role to develop a meaningful queue under increasing workload or complexity, which is plausible given that program managers are primarily attached to their own program.

The capacity and prioritisation experiments further show that adding internal capacity or changing dispatch rules would not materially shorten lead times at the current scale. Prioritisation rules produce almost identical outcomes under the historical workload, and only under stressed workload does the minimum-complexity rule provide a small improvement. This suggests that the relevant question is not which dispatch rule should be used immediately, but when the system moves into a regime where internal capacity and queueing begin to matter.

The forecasting application demonstrates how the digital shadow can be initialized from a live RVB data snapshot to generate project-level completion-time distributions and deadline-risk alerts. This provides practical value as an uncertainty-aware risk-screening tool: program managers can identify projects with elevated delay risk and inspect P20, P50, and P80 completion estimates instead of relying on single deterministic dates. However, because the model has not yet undergone formal interval-coverage validation and shows a bias in absolute lead time, the current forecasting output should be interpreted as a relative risk indicator rather than as a fully calibrated prediction instrument.

Overall, this thesis shows that a digital shadow can be built from data already available within a large public real estate organisation and used to analyse program execution dynamics. Its main value is not that it produces a new optimal scheduling rule, but that it reveals which mechanisms currently matter. For RVB, the central lesson is that

improving program predictability requires attention to the processes that make projects wait, rather than only to the internal roles that perform registered project work.

6.2 Answering the sub-questions

SQ1 — Required data and methods. A program-level digital shadow can be developed from milestone records, building metadata, project–building mappings, and hours registration data. These sources were transformed into project arrivals, project-level building characteristics, complexity scores, role-effort estimates, passive-delay parameters, and program-specific resource capacities. Together, these elements provide the empirical foundation for a discrete-event simulation model of program execution.

SQ2 — Resource allocation and prioritisation. At the historical workload, resource allocation and prioritisation have only limited effects on program performance. Role capacity is far from saturation, and the tested dispatching rules produce nearly identical outcomes. Only when workload is increased does prioritisation begin to matter, with the minimum-complexity rule producing a small improvement. This shows that internal allocation decisions become relevant mainly when the system enters a more congested regime.

SQ3 — Complexity, predictability, and variability. Project complexity increases active effort requirements, but its effect on total lead time is modest because passive delay dominates the overall duration. Program-level predictability is therefore not primarily determined by the complexity mix of the portfolio. Complexity is most useful for risk stratification and for identifying where coordination pressure may increase, particularly around the program manager role.

SQ4 — Predicting execution times. The digital shadow can generate probabilistic completion-time estimates from a live data snapshot. These outputs provide P20, P50, and P80 completion dates and deadline-risk indicators for active projects. However, because absolute lead time is biased and formal forecast interval validation has not yet been completed, the current forecasting application is best interpreted as an uncertainty-aware risk-screening tool rather than as a fully calibrated date-prediction system.

6.3 Recommendations for RVB

Based on the findings, three practical recommendations can be made for RVB.

First, RVB should focus improvement efforts on passive-delay drivers rather than only on internal role capacity. The experiments show that internal capacity is not currently the dominant constraint. Shortening lead times therefore requires better visibility and control over approvals, handovers, contractor lead times, regional execution phases, and administrative waiting between milestones.

Second, RVB should improve the linkage between parent projects and regional sub-projects. A key limitation of the current model is that hours written under regional sub-project numbers cannot always be attributed back to the originating project. This causes part of the active work to appear as passive delay. Recovering these links would improve both the accuracy of the model and the organisation's understanding of where project time is actually spent.

Third, RVB can use the digital shadow as a periodic risk-screening instrument. Even before full forecast calibration, the model can help program managers identify projects with elevated deadline risk, monitor the development of queues, and test whether changes in workload or prioritisation would materially affect program performance. This supports a move from reactive reporting towards simulation-supported program management.

6.4 Future work

The most important future improvement is to recover the relationship between parent projects and regional sub-projects. If regional hours can be reattributed to the correct parent project and milestone transition, the separation between active effort and passive delay can be improved. This would make the model more strongly effort- and capacity-driven and would clarify which part of the observed waiting time is genuine external delay and which part is unobserved internal work.

A second direction is formal forecast calibration. Future work should run the digital shadow from multiple historical snapshot dates and compare the predicted completion intervals with realised OAB dates. This would make it possible to measure prediction interval coverage, quantify forecast bias, and calibrate the deadline-miss probabilities before operational use.

Third, passive delay should be modelled in more detail. In the current model, passive delay is estimated as a residual. A more advanced version could explicitly model permitting, approvals, contractor lead times, handovers, and regional execution phases where data are available. This would improve the diagnostic value of the model by showing which waiting processes contribute most to program lead time.

Fourth, the scenario design could be extended. Replacing replayed PO dates with a calibrated stochastic arrival process would allow the model to analyse counterfactual intake policies. Similarly, replacing clone-based workload scaling with a generative model of project diversity would provide a more realistic view of how capacity constraints emerge under future portfolio growth.

Finally, the complexity model could be strengthened through additional expert validation and sensitivity analysis. Testing alternative AHP weights or comparing expert-derived weights with data-driven weighting schemes would show how robust the complexity score is and how strongly it influences the simulation outcomes.

6.5 Closing remark

This thesis demonstrates that large-scale public real estate program execution can be represented through a process-oriented digital shadow using data already available within the organisation. By linking passive project records to generative simulation, the model moves beyond descriptive reporting and enables mechanism exploration, scenario analysis, and risk screening. The key insight for RVB is that program performance is currently shaped less by internal capacity shortages than by the waiting time embedded in organisational dependencies, handovers, approvals, and contractor processes. Improving predictability therefore requires not only better scheduling, but also better visibility and control over the processes that make projects wait.

Chapter 7

Ethics and Data Management

A new requirement for the thesis is that there must be a short section in which you reflect on the ethical aspects of your project. This requirement is related to one of the final objectives that a graduated student of the Master of Computational Science must meet: "The graduate of the program has insight into the social significance of Computational Science and the responsibilities of experts in this field within science and in society". You don't need to devote an entire chapter to this; a short section or paragraph is sufficient.

I acknowledge that the thesis adheres to the ethical code (https://student.uva.nl/en/topics/ethics-in-research) and research data management policies (https://rdm.uva.nl/en) of UvA and IvI.

The following table lists the data used in this thesis (including source codes). I confirm that the list is complete and the listed data are sufficient to reproduce the results of the thesis. If a prohibitive non-disclosure agreement is in effect at the time of submission "NDA" is written under "Availability" and "License" for the concerned data items.

Short description (max. 10 words)	**Availability (e.g., URL, DOI)**	**License (e.g., MIT, GPL, Creative Commons)**
Example dataset 1	<github url>or Figshare	GPL
Example source code	DOI (from Zenodo)	MIT
Example sensitive data	NDA	NDA

Appendix A

Appendix

A.1 Role label translations

The role labels in the original RVB hours registration data are recorded in Dutch. To improve readability, Table A.1 provides the mapping between the original Dutch source labels and the English labels used in this thesis.

Table A.1: Dutch source labels and English role labels used in the thesis

Dutch source label	**English label used**
Projectmanagement	Project Management
Projectleider	Project Leader
Bouwkunde	Building Engineering
Elektrotechniek	Electrical Engineering
Werktuigbouwkunde	Mechanical Engineering
Objectmanagement	Object Management
Brandveiligheid	Fire Safety
Bouwkosten	Building Costs
Bouwfysica	Building Physics
Lijnmanagement	Line Management
Technisch beheer	Technical Management
Overige	Other

Bibliography

[1] Andrés Tello and Viktoriya Degeler. Digital twins: An enabler for digital transformation. Report, Groningen Digital Business Centre (GDBC), University of Groningen, June 2021. URL https://www.rug.nl/gdbc/.

[2] Jan Molter, Max Eichenwald, and Rainer Müller. The digital twin - a production-related review. *Procedia CIRP*, 128:418–423, 2024. doi: 10.1016/j.procir.2024.03.021.

[3] Werner Kritzinger, Matthias Karner, Georg Traar, Jan Henjes, and Wilfried Sihn. Digital twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine*, 51(11):1016–1022, 2018. doi: 10.1016/j.ifacol.2018.08.474.

[4] Hao Wang, Yisha Pan, and Xiaochun Luo. Integration of bim and gis in sustainable built environment: A review and bibliometric analysis. *Automation in Construction*, 103:41–52, 2019. ISSN 0926-5805. doi: 10.1016/j.autcon.2019.03.005.

[5] Rafael Sacks, Ioannis Brilakis, Ergo Pikas, Haiyan Sally Xie, and Mark Girolami. Construction with digital twin information systems. *Data-Centric Engineering*, 1 (e14):1–25, 2020. doi: 10.1017/dce.2020.16.

[6] Paulina Duch-Zebrowska and Katarzyna Zielonko-Jung. Integrating digital twin technology into large panel system estates retrofit projects. *Urban Planning*, 6(4): 164–171, 2021. doi: 10.17645/up.v6i4.4464.

[7] Carlos Henrique dos Santos, José Arnaldo Barra Montevechi, José Antônio de Queiroz, Rafael de Carvalho Miranda, and Fabiano Leal. Decision support in productive processes through des and abs in the digital twin era: a systematic literature review. *International Journal of Production Research*, 59(24):7404–7424, 2021. doi: 10.1080/00207543.2021.1898691.

[8] Manuel Jungmann, Timo Hartmann, Rahul Tomar, and Lucian Ungureanu. A combined digital twin and location-based management system. In *Proceedings of the 31st Annual Conference of the International Group for Lean Construction (IGLC31)*, pages 1267–1278, 2023. doi: 10.24928/2023/0184.

[9] Mariam Gómez Sánchez, Eduardo Lalla-Ruiz, Alejandro Fernández Gil, Carlos Castro, and Stefan Voß. Resource-constrained multi-project scheduling problem: A survey. *European Journal of Operational Research*, 309(3):958–976, 2023. doi: 10.1016/j.ejor.2022.09.033.

[10] Deanna M. Kennedy, Sara A. McComb, and Ralitza R. Vozdolska. An investigation of project complexity's influence on team communication using monte carlo simulation. *Journal of Engineering and Technology Management*, 28(1-2):109–127, 2011. doi: 10.1016/j.jengtecman.2011.03.001.

[11] Mehran Barani Shikhrobat, Roger Flanagan, and Shabnam Kabiri. Understanding complexity at the pre-construction stage of project planning for construction projects. *American Journal of Operations Research*, 15(1):1–27, 2025. doi: 10.4236/ajor.2025.151001.

[12] Long D. Nguyen, Dai Q. Tran, An T. Nguyen, and Long Le-Hoai. Computational model for measuring project complexity in construction. *2016 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 1111–1115, 2016. doi: 10.1109/IEEM.2016.7798051.

[13] Ming Lu. Simplified discrete-event simulation approach for construction simulation. *Journal of Construction Engineering and Management*, 129(5):537–546, 2003. doi: 10.1061/(ASCE)0733-9364(2003)129:5(537).

[14] Félix Villafáñez, David Poza, Adolfo López-Paredes, Javier Pajares, and Ricardo del Olmo. A generic heuristic for multi-project scheduling problems with global and local resource constraints (rcmpsp). *Soft Computing*, 23:3465–3479, 2019. doi: https://doi.org/10.1007/s00500-017-3003-y.

[15] Serghei Floricel, John L. Michela, and Sorin Piperca. Complexity, uncertainty-reduction strategies, and project performance. *International Journal of Project Management*, 34(7):1360–1383, 2016. doi: 10.1016/j.ijproman.2015.11.007.