UNIVERSITY OF PANNONIA

DOCTORAL (PHD) THESIS

Bridging theory and practice: simulation-based scheduling performance evaluations for Application Lifecycle Management

DOI:10.18136/PE.2024.896

Author:
Róbert JAKAB

Supervisor:
Prof. Dr. habil. Zsolt Tibor KOSZTYÁN

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in the

Doctoral School in Management Sciences and Business Administration
Department of Quantitative Methods

August 26, 2024

Bridging theory and practice: simulation-based scheduling performance evaluations for Application Lifecycle Management

Thesis for obtaining a Ph.D. degree in the **Doctoral School in Management Sciences and Business Administration** of the University of Pannonia

in the field of **Social Sciences**
in the subject of **Management and Business Studies**

Written by: Róbert JAKAB
Supervisor: Prof. Dr. habil. Zsolt Tibor KOSZTYÁN

Propose acceptance (yes/no) ....
Supervisor

As a reviewer, I propose acceptance of the thesis:

Name of reviewer: .... yes / no

....
(Reviewer)

Name of reviewer: .... yes / no

....
(Reviewer)

The Ph.D. candidate has achieved ....% at the public discussion.

Veszprém, ...
(Chairman of the Committee)

The grade of the Ph.D. Diploma ....(........%)

Veszprém, ...
(Chairman of UDHC)

UNIVERSITY OF PANNONIA

Abstract

Doctoral School in Management Sciences and Business Administration
Department of Quantitative Methods

Doctor of Philosophy

Bridging theory and practice: simulation-based scheduling performance evaluations for Application Lifecycle Management

by Róbert JAKAB

Over the past decades, software has become an essential enabler for science and the economy. The evolution of software application development and maintenance handling has become an important domain both in academia and in business practice. In the SW development management from the one-time linear development approach, the focus moved to agile, flexible content handling and regular SW upgrade approaches. Several vendors are providing tools and toolsets supporting lifecycle development, supporting the concept of software life cycle, however, the related academic literature is still scarce in the area, such as availability of clear definition, methodologies and methods.

This dissertation begins with a thorough systematic literature study to identify and provide ALM attributes. Additionally, it aims to establish a comprehensive definition to facilitate future methodological research. The primary objective of this work is to assess the efficiency of scheduling algorithms in the ALM domain, considering the traditional, agile, and hybrid project management approaches, using a simulation-based model. Finally, a business case study illustrates the difficulty encountered in a recent ALM environment, conducted in an automotive supply company.

UNIVERSITY OF PANNONIA

Zusammenfassung

Doctoral School in Management Sciences and Business Administration
Department of Quantitative Methods

Doctor of Philosophy

Bridging theory and practice: simulation-based scheduling performance evaluations for Application Lifecycle Management

von Róbert JAKAB

Software hat sich in den letzten Jahrzehnten zu einem wesentlichen Enabler für Wissenschaft und Wirtschaft entwickelt. Die Entwicklung der Softwareanwendungsentwicklung und der Wartungsabwicklung ist sowohl in der Wissenschaft als auch in der Geschäftspraxis zu einem wichtigen Bereich geworden. Im SW-Entwicklungsmanagement verlagerte sich der Fokus vom einmaligen linearen Entwicklungsansatz hin zu agilem, flexiblem Content-Handling und regelmäßigen SW-Upgrade-Ansätzen. Mehrere Anbieter bieten Tools und Toolsets zur Unterstützung der Lebenszyklusentwicklung an, die über das Konzept des Software-Lebenszyklus hinausgehen. Allerdings ist die entsprechende wissenschaftliche Literatur im Bereich klarer Definitionen, Methoden und Methoden noch rar.

Diese Dissertation beginnt mit einer gründlichen wissenschaftlichen Literaturstudie zur Identifizierung von ALM-Attributen. Darüber hinaus soll eine umfassende Definition erstellt werden, um zukünftige methodische Forschung zu erleichtern. Das Hauptziel dieser Arbeit besteht darin, die Effizienz von Planungsalgorithmen im ALM-Bereich unter Berücksichtigung der traditionellen, agilen und hybriden Projektmanagementansätze mithilfe eines simulationsbasierten Modells zu bewerten. Abschließend veranschaulicht eine Geschäftsfallstudie die Schwierigkeiten, die in einer aktuellen ALM-Umgebung, die in einem Automobilzulieferunternehmen durchgeführt wurde, aufgetreten sind.

Acknowledgements

Professional acknowledgements

I express immense gratitude to my supervisor, Prof. Dr. habil. Zsolt Tibor Kosztyán, for his unwavering dedication and inexhaustible mentorship during these years. I would like to express my gratitude to my co-authors during the last year's publications and the members of the methodological research group, as well as my colleagues at the company, for their valuable contributions and encouragement.

Personal acknowledgements

I am grateful to my family, particularly my parents and sister, who consistently believed in me and encouraged me. I also appreciate my wife for her endless patience and unconditional love. Special thanks to my friend Gergely Novák, for being a great professional help above our friendship.

Finally, I appreciate all the valuable feedback and discussions from peer reviewers, proofreaders, and experts.

"Impossible is nothing!" - Muhammad Ali, 1974

Contents

List of Figures

List of Tables

List of Abbreviations

In alphabetical order:

ABM	**A**gent **B**ased **M**odeling
AHP	**A**nalytical **H**ierarchy **P**rocess
ALM	**A**pplication **L**ifecycle **M**anagement
ANOVA	**AN**alysis **O**f **VA**riance
AoA	**A**ctivity **o**n **A**rrow
AoN	**A**ctivity **o**n **N**ode
API	**A**pplication **P**rogram **I**nterface
APM	**A**pplication **P**ortfolio **M**anagement
APM	**A**gile **P**roject **M**anagement
APMa	**A**gile **P**roject **M**anagement **a**gent
BOM	**B**ill **o**f **M**aterial
CAx	**C**omputer **A**ided x (where, x can be e.g., D - Design)
CCTA	**C**entral **C**omputer and **T**elecommunications **A**gency
CD	**C**ost **D**omain
CIO	**C**hief **I**nformation **O**fficer
CR	**CR**iterion or **C**hange **R**equest
CSV	**C**omma-**S**eparated **V**alues
DevOps	**Dev**elopment and **Op**erations
DMM	**D**omain **M**apping **M**atrix
DSDM	**D**ynamic **S**ystems **S**evelopment **M**ethod
DSM	**D**esign **S**tructure **M**atrix
EF	**E**arliest **F**inish
ERP	**E**nterprise **R**esource **P**lanning
ES	**E**arliest **S**tart
EST	**E**arliest **S**tart **T**ime
GA	**G**enetic **A**lgorithm
GS	**G**oogle **S**cholar
HPM	**H**ybrid **P**roject **M**anagement
HPMa	**H**ybrid **P**roject **M**anagement **a**gent
ICT	**I**nformation and **C**ommunications **T**echnology
IS	**I**nformation **S**cience
IT	**I**nformation **T**echnology
ITIL	**I**nformation **T**echnology **I**nfrastructure **L**ibrary
LD	**L**ogic **D**omain
MAS	**M**ulti-**A**gent **S**ystem
MCS	**M**onte-**C**arlo **S**imulation
MMLIB	**M**ulti-**M**ode **lib**rary
MNA	**M**eta-**N**etwork **A**nalysis
ND	**N**on-renewable resource **D**omain
NPD	**N**ew **P**roduct **D**evelopment
MRCPSP	**M**ulti-mode **R**esource **C**onstrained **P**roject **S**cheduling **P**roblem
MRCMPSP	**M**ulti-mode **R**esource **C**onstrained **M**ultiproject PSP

OCG	**O**ffice of **G**overnment **C**ommerce
OEM	**O**riginal **E**quipment **M**anufacturer
OSLC	**O**pen **S**ervices for**L**ifecycle **C**ollaboration
PDM	**P**roduct **D**ata **M**anagement
PDM	**P**roject **D**omain **M**atrix
PEM	**P**roject **E**xpert **M**atrix
PLC	**P**roject **L**ife **C**ycle
PLM	**P**roduct **L**ifecycle **M**anagement
PM	**P**roject **M**anager or **M**anagement
PMI	**P**roject **M**anagement **I**nstitute
PMO	**P**roject **M**anagement **O**ffice
PRISMA	**P**referred **R**eporting **I**tems for **S**ystematic Reviews and **M**eta-**A**nalyses
PSP	**P**roject **S**cheduling **P**roblem
PSPLIB	**P**roject **S**cheduling **P**roblem **LIB**rary
QD	**Q**uality **D**omain
RA	**R**esearch **A**ssumption
RAD	**R**apid **A**pplication **D**evelopment
RCPSP	**R**esource-**C**onstrained **P**roject **S**cheduling **P**roblem
RCPSP-AC	RCPSP with **A**lternative activity **C**hains
RCPSP-AS	RCPSP with **A**lternative **S**ubgraphs
R&D	**R**esearch and **D**evelopment
RD	**R**enewable resource **D**omain
RFS	**R**andom **F**orest **S**urvival
RUP	**R**ational **U**nified **P**rocess
RQ	**R**esearch **Q**uestion
RT	**R**esearch **T**hesis
SABRE	**S**urvival **A**nalysis-**B**ased **R**isk **E**valuation
SAFe	**S**caled **A**gile **F**ramwork **e**nterprise
SDLC	**S**oftware **D**evelopment**L**ife**C**ycle
SDP	**S**oftware **D**evelopment **P**roject
SLA	**S**ervice **L**evel **A**greement
SLR	**S**ystematic **L**iterature **R**eview
SOP	**S**tart **O**f **P**roduction
SST	**S**cheduled **S**tart **T**ime
SW	**S**oft**w**are
TD	**T**ime **D**omain
TPC	**T**otal **P**roject **C**ost
TPM	**T**raditional **P**roject **M**anagement
TPMa	**T**raditional **P**roject **M**anagement **a**gent
TPQ	**T**otal **P**roject **Q**uality
TPR	**T**otal **P**roject **R**esource
TPS	**T**otal **P**roject **S**core
TPT	**T**otal **P**roject **T**ime
xPMa	**x** **P**roject **M**anagement **a**gent
XPM	e**X**treme **P**roject **M**anagement
WBS	**W**ork **B**reakdown **S**tructure
WIP	**W**ork **I**n **P**rogress

List of Symbols

Latin symbols

$a_i(T)$	scheduled execution time interval of task a_i
a_i	task i
$f\%$	rate of flexible dependencies
fp	flexibility parameter, the ratio of flexible dependencies and prioritized tasks to all tasks and dependencies
k	number of task completion modes
$l_{ij} = [\mathbf{LD}]_{ij}$	element of the logic domain, task occurrence if $i = j$, and arc that represent the precedence relation between tasks i and $j \neq i$ (in this case, $l_{ij} = 1$ means task i precedes task j)
n	number of tasks
n'_L	number of arcs with length L
m	maximal number of progressive levels
$\overrightarrow{P}$	task path (sequence)
P_i	set of immediate predecessors of task i
r_{ij}	demand of task i for renewable resource type j
$r_{ij}(\tau)$	demand of task i for renewable resource type j at time τ
$s\%$	rate of supplementary tasks
S_i	set of immediate successors of task i
t_i	duration of task a_i
w_i	width of progressive level i, $i = 1, ..., m$

Greek symbols

α_j	availability of renewable resource type j
α_w	total absolute deviation from the average width
η	number of types of nonrenewable resources
ρ	number of types of renewable resources

Calligraphic symbols

$\mathcal{A}$	set of arcs (dependencies)
$\lvert\mathcal{A}\rvert$	number of dependencies in a project structure
$\mathcal{S}$	project structure, set of (to-be-) realized tasks
$\overrightarrow{\mathcal{S}}$	project schedule of project structure $\mathcal{S}$

Chapter 1

Introduction

The chapter Introduction presents information on the ALM subject including its historical background, characteristics, and challenges. After that, the gaps and novelty of the area are revealed. The section on Research goals outlines the overarching objectives that will guide the subsequent section, which focuses on the specific Research Questions for this dissertation. The chapter concludes by presenting the structure of the dissertation to the reader.

1.1 ALM historical background

In today's fast-expanding technology landscape, the increasing reliance on software programs as a cornerstone of modern corporate operations highlights the requirement to understand and manage their development efficiently (Hofacker, 2019; Rokade, 2008; Singh and Ahlawat, 2023).

It is essential to address the challenge of the rapidly changing environment and market demands in product development recently. Companies nowadays need to adapt their thinking and working style to be more oriented toward flexibility rather than relying on rigid, strategic approaches like in previous decades. Formerly, for general project approaches, there have been several decades and opportunities in many industries to develop and adapt proper methodologies to increase output and improve efficiency. This has been a long-term evolution that was based on strategic adaptation and standardization to manage the lifecycle of products. Recently, these adaptation processes have been forced into shorter cycles and extended in scope, with increased complexity. The management of software and software applications are becoming more and more integral parts of essential product development. Achieving comprehensive control of this new situation requires further specific knowledge and competency. Focus on this is particularly important, as the distinction between general projects and software projects has been evident in recent decades.

The challenge of managing software within the project framework has been prevalent since its inception. Since the 1990s, when the Standish Group published the first CHOAS study (Clancy, 1995), the success rate of projects has been considerably lower compared to typical construction projects. Software projects differ from traditional projects, such as construction projects, in various aspects. These differences include their structure, intangible results, typically higher complexity, unique design and documentation requirements, and the management of their product's lifecycle, among other features as the study highlights. According to the above-referenced research, it was not successful enough to address these disparities effectively using the well-known traditional approaches and methods. As a result, this phenomenon has captured the attention of the scientific and expert communities, who have since then actively studying and enhancing the features of software development. They are eager to continuously improve their tools in order to intensify execution and achieve more efficient developments, ultimately leading to improvements. Therefore, in project management where software-related activities are involved, three main development trends emerged: *traditional, agile and hybrid*.

The traditional approach strongly relies on clear requirements availability upfront and following a well-defined schedule and resource distribution from the planning phase already. Major changes are not warmly welcome in the downstream phase; however, upstream is still possible, with significant adaptation efforts. Potential delays and cost overheads are therefore highly likely. This approach is potentially fitting for well-defined application development with low risk of changes, however, it will get in trouble in case the requirements are frequently changing and adaptations in plans are often necessary. Also, where there is sensitivity for cost and timing by the customers, this approach is not a perfect fit. These attributes, however, are highly distinctive for software application development.

After several years of small adaptations and modifications of this traditional approach has not resulted in a breakthrough in efficiency increase, a fundamentally new approach was established by experts in the software field, and established the so called agile software development principles in 2001 (SGI, 2019). The naming refers to the ability of quick adaptability, as the agile development's main focus is the current customer requests and their involvement in the development. Here, the main target is clarified, and instead of focusing on a pre-defined fixed schedule, this approach can work out with several, iterative planning alignments within a flexible structure, where the goals are broken down into smaller tasks with their fixed iterative loops of development. The small fixed time periods (called sprints) with fixed content keep the rhythm and pace of development with the schedule and content adaptability option at these specific events. So despite the common belief that the agile method does not contain planning, it is on the contrary, several fixed timeslot plannings are taking place, involving several levels from engineering to management, which encourages stakeholder involvement and communication with management. By the way, these were also named as critical factors for success in the CHAOS report previously. Agile team is required to be fully dedicated to the given project, sharing resources is possible but not welcome. This approach is best fitting for single-project applications, however, scaling of agile projects is also possible (Knaster, 2023).

In the hybrid approach, the traditional and agile methods are combined, enabling new activities can appear and be involved anytime, and also capable of handling multiple projects at once. These approaches' applicability and efficiency will be studied in this dissertation specifically.

Businesses of all sizes are leveraging software applications to innovate, streamline processes, and give value to customers. Consequently, the economic impact of software creation and maintenance has expanded dramatically over the decades as this meant economic and competitive advantage (van den Ende and van Marrewijk, 2014; Al-Saqqa et al., 2020; Mishra, 2023).

The general purpose of the present dissertation is also to contribute to the evolution of SW application development efficiency, more precisely in the application management area.

Technically and methodological perspective software development has a significant history already. In the early days of computer science (1950s), when operating systems were not even developed, each program required a complete hardware specification in order to function properly and carry out basic functions. The increasing complexity of hardware and their embedded application programs meant overhead and difficulties in case of even the slightest changes. It finally required the separation of software and hardware to enable their economical, quicker improvement. Since the hardware and software decoupling first happened for personal computers in the 1960s (Guendert, 2011), the advantage of this approach for user-oriented software program development started to flourish. More and more software application development started and took place, primary in the IT industry and later also in areas where software-controlled equipment appeared. Managing the SW through its lifecycle, which involves the inception of the idea, through the design, development, release and operation, and even the retirement of the software several methods and approaches worked out, however, mostly specific and limited to the specific phases. Some were based on the classical, traditional approaches some were completely new, reform ideas like agile development in the 2000s. However, the focus idea was always to deliver the best possible software solution.

Apparently, a similar trend was visible not only in the IT area, but also happened later for mobile telecommunication in the 2000s, when users demanded devices to be capable of configuring according to their needs. Being able to install and remove applications easily according to their preferences.

Similarly, as software became a defining factor also of modern vehicles, an increasing number of users anticipate that these features will be configurable and regularly updated, in the way applications are updated on their smartphones. Therefore, recently in automotive, such a trend is appeared, not even only for comfort but also for safety functions as well with a so-called software-defined-vehicle concept. SW updates for the vehicles are more and more common, and not only in the service stations but even wireless automated updates via the internet (Haar, 2021; Resing, 2023).

Therefore, the failure of software projects can be economically disastrous. Delays, budget overruns, inefficient content and change management, and sub-optimal software quality are not only costly but also erode market competitiveness. The meticulous management of the software applications is essential for mitigating these risks and maximizing the economic efficiency of software development projects through the entire software application journey, and ensuring that investments in software translate into sustainable economic returns. That's why in the present era, within the IT industry, there is a shift in the need for software application development from the conventional approach of one-time, linear development to handling the constant need for upgrades.

These are the reasons why, in the industry, the challenge for successful software application management is present, and finding improvements in the area is vitally important. This dissertation is also contributing to the evolution of this area, namely the ALM performance evaluation and sharing the results with the area experts and academic community.

As the management of the lifecycle of products is already known in project management literature as Product Lifecycle Management (PLM), the obvious approach was to apply it to the software product also. However, the differences between physical and software product development, such as change management include the control and handling of changes in requirements, architecture, design models, source code, documentation, configuration data, test cases, and other software-related elements, forced severe compromises and tailoring of PLM approach, still not excluding though a potential later synergy again (Krueger, 2015).

Application Lifecycle Management (ALM) has emerged in this era as a crucial focal point for enterprises across numerous industries and sectors including information technologies, automotive, healthcare, and aerospace just as recent examples. It is observable in the last few decades an unparalleled transformation towards a software-centric economy (Andreessen, 2011). ALM focuses on the entire lifecycle of the software or application, while classical product lifecycle management (PLM) is mostly related to physical product development. Artifacts and deliverables for a SW and a physical product are also different and need different environment integration and handling methods (Deuter and Rizzo, 2016). Thus, adaptation in classical project management is also required for such endeavors. Traditional and well-known methodologies are visibly underperforming and are no longer suitable or effective for use in this altered context when unexpected or additional task management is required. The topic concerns the discussion of a recent scheduling difficulty, specifically focusing on the weaknesses of currently existing project methodologies from a methodological standpoint. In standard linear execution, there is no prioritization, while there is the possibility of multiple execution modes of tasks (e.g., using alternative technology or different approaches to carry out the activity, usually with a trade-off between resources and time) target solution. In contrast, for agile execution, it is possible to prioritize tasks and rearrange the execution order, but there is only one mode to target. The hybrid methodology has the capacity to prioritize and allow for multimode execution. To obtain an organized summary, please refer to Table 3.2 in the section 3.2 Agent based implementations of the dissertation.

From practice, can be seen also, that for companies, it is a recent challenge to efficiently initiate application lifecycle management (ALM), which is due to three main factors. Defining ALM is challenging because of the complex interrelationships among many lifecycle activities, including the product, project, staff, procedures, tools, and technology. Furthermore, ALM tasks require tools that are specifically tailored to meet their needs. Lastly, efficiently carrying out ALM operations requires a significant level of discipline (Cheng, 2010). It is clear from these challenges that further support from academia is needed to increase efficiency, as tools are providing an environment, but not necessarily a solution for scheduling for example.

The primary objective of academic work is to assist decision-makers, particularly in the business sector. The objective is to address intricate issues, and economic phenomena to optimize execution, minimizing expenses to enhance profitability. Thus, studying the effectiveness of the approach of Application Lifecycle Management (ALM), which encompasses several components such as communication and coordination, process and visibility, traceability for compliance, access management, milestone checks, feasibility analysis, and tasks execution planning throughout the application's lifecycle (Magid, 2007) can result in improvement of SW application development in several aspects.

1.2 Research gaps and novelty of area

It is vital that ALM is acknowledged as more than just a technical need; rather, it should also be recognized as an economic phenomenon or a business challenge. This dissertation is driven by this imperative also. The author met the ALM problem at an automotive supplier company, which is focused on safety-critical software development for brake controllers. The challenge appeared when the setup changed from the customer side and requested post-deployment updates several times however, in an unexpected distribution. The formerly used traditional project management method was not efficiently working out anymore, several times the new requests caused delays in the original program, kept the resources bound causing higher costs and cost overruns, and the ad-hoc releases scheduling in the normal system was also meeting obstacles to meet requested customer milestones, resulting also additional escalations and extra work. The irregularity in the requests and extended scope without a fixed finish date made the traditional project approach extremely inefficient and management started to look for a proper planning and scheduling solution. The author was involved in this process from the project management lead side and actively participated in the elaboration of new project management approaches, like agile practices, and introduction for the company. The configuration similarities and realization happened during the research of the academic literature and the parallel studies led in the university research group side. The case study provides more details about the relevant business problem and its solution.

Therefore high level purpose of this work is to investigate scheduling feasibility methods that are able to support and improve ALM scheduling execution. As a result, to fulfill the purpose of this dissertation is to present an academic and exploratory overview first of the concept of ALM. This is essential in order to show the context of ALM in the academic understanding. Important to notice, that next to academic sources, there is even a stronger non-academic, business-driven written works for ALM existing. These are serving mainly the vendors' marketing and communication strategies, to promote their understanding and tools for their potential customers. The focus of this work is on academic and peer-reviewed sources, as predominantly, software vendors assume the stronger role compared to academics, leading to a lack of thorough research and deficiencies in scientific basis in several domains, particularly in scheduling, which can significantly impact company outcomes from an economic perspective.

The first research gap in the ALM field has already been noticed by the search for ALM definitions. Due to the relatively recent and intricate nature of ALM, it was evident during the preliminary analysis that there is a large presence of commercially-driven literature and a scarcity of academic sources in comparison. Additionally, the rapid evolution of tools and software development processes has also had an enormous impact on the field. Therefore, academics may also differ in their definition of scope. Some authors draw strong parallels between ALM and PLM, asserting that ALM is only software-relevant PLM (Ebert, 2013; Deuter, Otte, et al., 2019). Others claim that ALM has a significantly wider scope (Kääriäinen, 2011). The similarities like both ALM and PLM are multi-disciplinary approaches and the differences like PLM is generally a physical product-related method, meanwhile, ALM is software centric, and more details are presented in Subsection 2.3.1 comparing the PLM and ALM approaches. However, a comprehensive and thorough literature analysis on the concept of ALM has not yet been conducted to offer any insights into the existing definitions proposed by researchers. This dissertation aims to address this gap using a cross-sectional Systematic Literature Review. Additionally, it will provide an overview and categorization of the ALM definitions based on a Critical Review.

After gaining an awareness of the ALM environment, the next possible step is the evaluation of scheduling performance in the ALM environment. Currently, there is not yet a significant amount of research being conducted to address this second gap. There might be several reasons for it, which are due to the novelty of this area, for example, it exists in a multi-disciplinary area where a methodological evaluation target is a challenging endeavor by default. But also the novelty of the ALM environment with its scattered or interrupted development which is facing a challenge to traditional linear thinking, see Figure 2.8 (Chappell et al., 2010; Rossman, 2010). The dissertation's objective was to conduct an analysis of project management strategies that already exist, including traditional, agile, and hybrid approaches, to determine their application and efficiency in ALM environment. Usually addressing methodological approaches in ALM area are related to agile methodology (Knaster, 2023; Ciric, Lalic, et al., 2018) or comparison between traditional and agile methods, however, they mostly stuck on theoretical or survey level highlighting the advantages of agile (Ciric, Lalic, et al., 2019; D. Fernandez and J. Fernandez, 2008), very few goes to case study and measured performance evaluation (Wells et al., 2015; Vresk et al., 2020), even in ALM area hard to find any similar efforts taken (Kääriäinen and Välimäki, 2008; Tüzün et al., 2019). Important to notice that even tough agile is praised mostly against the traditional methods, there are drawbacks also for agile methods, which for specific areas can be disadvantageous. Enough to think about the content fulfillment flexibility from customer perspective or dedicated team requirements from supplier point of view (Gumiński et al., 2023; DŽANIĆ et al., 2022). The scope of the examined scheduling methods is limited to only these three variants only due to the fact that the traditional and agile are the most commonly used and examined methodologies in the SW development area (Clancy, 1995), and also in practice either one of them or their combination, the hybrid method is used. Needless to say, further scheduling methodologies and processes (Kanban, Extreme Programming, Lean Development, Critical Path Method, Program Evaluation and Review Technique, Feature Driven Development, Rapid Application Development, Spiral Model, etc.) are worth examination in the future research steps, however, due to limitation of thesis scope those are omitted now.

However, as in practice and in an economic environment that is constantly changing, it is becoming increasingly important to identify improved solutions for the development of SW-products-related like ALM, as seemingly the traditional or waterfall method is underperforming in a rapidly changing environment, handling changes for example, in a more complicated way than agile. The examination of such sensitivity is interesting to look at, and such analysis is also proceeded in this dissertation.

So this dissertation addresses the gap in both theoretical (definition research) and practical (feasibility and methodology analysis) aspects also. By doing so, these results might be basis though for the later, further studies to develop new or adapt existing methods with higher performance, which can result improved economic results.

Knowing these high-level targets, the fitting research methods were identified. Targeting the studying the academic environment for ALM, leading a preliminary review revealed, that only a very limited number of scientific material is available and hardly any review articles are existing on the ALM field yet. Therefore for getting a comprehensive view the systematic literature review tool was selected to identify the ALM definition.

In the ALM scope, then, the question arises of how to adapt or extend these project scheduling methodologies, for example, to handle the additional tasks and see how they are within the original definition of understanding traditional, agile, or hybrid management. Thorough examination and evaluation are needed to determine the applicability conditions. This can happen with a literature review from the already existing academic literature and is also confirmed within the frame of expert discussion of the case study, as PLM and ALM have many differences, and project management tools applications are not obviously possible. The feasibility conditions thus needed careful checks and confirmations.

How to handle additional activities in the sense of boundaries of scope, possible to extend them? Does the client pay more for the newly adapted plans or is it included in the original contract? If included, how is it handled? Is there any definition of allowed content or are all new tasks to be handled as change requests? How can the extension support the initial boundary conditions, or is there flexibility included in the scope by default? How the risks are changing and accumulating by such ALM activities compared to the classical projects? As visible, several open questions arise getting deeper investigation of the topic. These were some guiding questions for a better understanding of the problem itself in the ALM scope. Therefore a controlled and limited environment, a simulation setup was selected to proceed with, to be able to focus on the main factors. The existing limitations are considered and adaptations to real-life data as best possible selected. However, simulations are always a constrained model of reality, with oftentimes a comfortable academic pace, thus additional validations were desirable to counter-check the results in real life. Thus, a case study was also conducted at a company where the ALM is a daily challenge, where the management together with the field experts are exploring ways of improvement with the pressure of business targets. A significant automotive electronic controller supplier company was open to proceeding and evaluating the findings with this theoretical approach.

1.3 Research goals

In this section the research goals are presented, which are basically the description of the high level goals of the dissertation, based on the previous section introduced gaps and novelty of the ALM area.

The ultimate aim of this dissertation is to analyze and evaluate the performance of the traditional, agile, and hybrid project management approaches in the application lifecycle management context.

Present thesis comprises three primary components. The first part to presents a comprehensive literature study aimed at exploring the ALM field and making a contribution to the existing literature. This is to be achieved by proposing a unified definition and critically reviewing the findings, while also including business experiences. The second part is to examine the project management approaches in the ALM environment, namely the traditional-, agile-, and hybrid-project management approaches. Their evaluation to be conducted using a simulation environment, where not only examining the scheduling performance was possible, but also a risk analysis for specific parameters. The third part is showing a case study about an ALM environment in an automotive electronic supplier company, where an agile way of working was also introduced. Expert and management interviews to support the modeling and evaluation. Doing so in an academic format are valuable contributions for the ALM literature.

Therefore, as a high-level summarization of the aims of the dissertation are the following:

1 Research of ALM scientific literature for

- definition and scope identification;
- enabling definition determination for methodological research.

2 To confirm the applicability of Matrix representation for scheduling investigation, including:

- simulation (artificial) environment setup,
- TPM, APM and HPM feasibility check,
- TPM, APM and HPM scheduling efficiency analysis.

3 Examine the effects of risk factors on the IT project's structure for scheduling.

Supporting the relevance of the research topic, conducting a relevant ALM case-study with scheduling performance evaluation.
The aims are precised in the following section Research questions.

1.4 Research questions

In this section, the research goals are further detailed into more detailed research questions below. The research questions are phrased to have a more focused, clear and researchable format compared to the more general research goals. In the dissertation the research questions are processed with literature review and theoretical background work based on them a research assumption will be in the later chapters elaborated. With the execution of the research of the points of the thesis, the assumptions are either accepted or rejected and the relevant research theses are expressed.

RQ1: How can a planning model be identified based on available scientific literature definitions that represents the Application Lifecycle Management (ALM) scheduling problem?

RQ2: Do the existing project management scheduling methodologies (TPM, APM, HPM) produce feasible solutions in the ALM environment, and how are they performing?

RQ3: What are the risk factors for the scheduling problem in the ALM environment, and how are they influencing the feasibility and scheduling performance?

In Section 6.6, there is a summary table that collects the Research Questions, Assumptions, and Theses for an overview.

For answering the **RQ1** a detailed and thorough literature review is providing the context for the ALM is necessary. The aimed contribution with these targets to broaden the knowledge of the ALM field for professionals and academics. Such summarizing review for the ALM definition was not yet available, thus this is a useful input for the field for business-related stakeholders to get a broader picture of the ALM content and understandings. This might help them to realize their situation and support business decisions for example for purchasing proper IT and development tool sets.

Related to the **RQ2** examination of the commonly used the project management approaches performance evaluation a highly expected contribution to the ALM field for scholars to identify and proceed with further research for example in schedule refinement methods development next to many other challenges present in the field. This could serve also later as an economic advantage for business decision makers to select the most fitting approach for their targets to realise. In the analysis the details of the project management approaches serves the better understanding for field experts, and by the evaluations with recommendations for each specific setups for business and development targets are present. Therefore the management decision makers can see and decide which approach is serving best their target such as time, cost, resource or customer satisfaction for their software application management during its lifetime.

Looking at the **RQ3** context, the relevant risk factors of the specific ALM field evaluation part can be particularly important and interesting for business stakeholders and field experts, as usually risk realization is connected to negative effects. Therefore the more prepared the better-handled rule is valid, which means recognizing and mitigating the risks in the early phase can have fewer effects on the plans. As application lifecycle management is a specific field even in the IT area, there are several factors whose relevance need to be examined, and also the ALM-specific risks need identification. This research is providing even a summary for comparison and identification of the main findings during the ALM area examination.

1.5 Structure of the thesis

This section is describing the structure of the thesis with the highlights and targets of the chapters.

The dissertation is relying on three main building blocks, in a sequential order they are building on each other as represented on the Figure 1.1 also.

In Chapter 2, starting with a brief overview of project management and an overall scope overview of Application Lifecycle Management, it is demonstrated a rigorous literature review to examine the definitions of ALM in the academic literature. Furthermore, a critical evaluation not only analyzes these aspects but also offers an original interpretation of ALM model to support future methodological studies. Following that, the logic planning and scheduling problem is introduced, with the project management methodologies (traditional, agile, and hybrid) and related risk handling background, which serves as the primary foundation of the dissertation's base.

Chapter 3 presents the research framework with the assumptions, the agent based implementations, the simulation. The introduction of data sources is followed by the presentation of the simulation environment.

The results of the literature review and the simulation are reported in Chapter 4 including the discussion about the results can be located here also as the second building block of the dissertation.

Chapter 5 serves as the third, last fundamental aspect, in which a recent case study is presented, focusing on the ALM problem in the automobile industry as validation of the results.

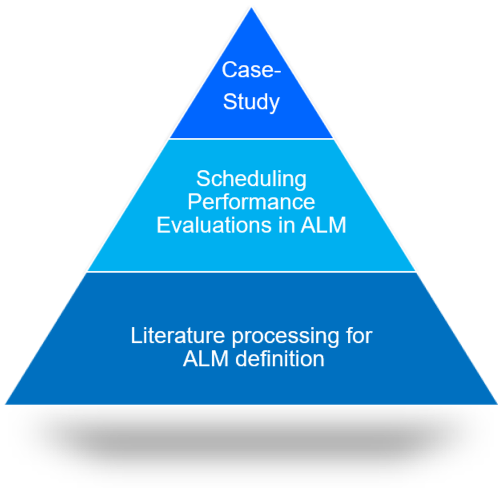


FIGURE 1.1: Pillars of this dissertation
(Source: own edit)

Chapter 6 provides a comprehensive summary and draws conclusions that address the research issues. Emphasizing the implications for both academia and practitioners as well.

Chapter 2

Literature review

This chapter offers a contextual and background information for the research scope of the thesis research topics. This includes summarizing, evaluating, and integrating prior research on the background of the ALM problem to provide the groundwork for the research questions related to the theses.

2.1 Projects and Project management

This chapter offers a brief description of project and project management, which purpose is to facilitate comprehension of the fundamental concepts and definitions that are essential for understanding the scheduling problem and serve as a foundation for comprehending the ALM problem domain in the next steps. Even though it is not covering fully the project and project management area, it includes, based on the available literature sources, the theoretical foundations of project and project management, a comparison of project management approaches, the characteristics of software projects and application projects, the theoretical and practical history of ALM, the concept, development and special features of ALM, ALM , SDLC, PLM relationship system, as well as the description of project management approaches (traditional, agile, hybrid) and their applicability in the case of software projects and application projects.

2.1.1 Project Definitions

Several efforts have been made to define projects in the academic and professional literature in the last decades (Whelton and Ballard, 2002; Cho and Gibson Jr, 2001; Morcov et al., 2020), however, there is not one unique overall definition existing. The project definition has evolved from traditional and rigid formulations to modern and flexible frameworks. Contemporary project definitions highlight the need for adaptability, involving all parties with an interest in the project, and continuously improving through a cyclical process. By employing approaches like Agile and Design Thinking, projects are designed in an iterative manner, allowing for the incorporation of evolving needs and feedback from stakeholders. This strategy promotes the capacity to adjust and react effectively in intricate and unpredictable situations. Furthermore, several modern definitions give importance to the precise expression of project goals, scope (Lock, 2020, p.33), and criteria for success (PMI, 2021; Meredith and Mantel, 2019), which helps to align the interests of all parties involved and improve the results of the project. Today's project definitions enable efficient planning and execution in dynamic organizational environments by embracing iterative and collaborative procedures. Also important to note that several scholars are respecting the project as temporary organizations, and have a different view on the definition approaches (Hansen et al., 2022), however as target of present thesis is focus on the methodological approach, thus more precise and detail oriented project definitions are favoured.

In the following some of the key project definitions follow, and their applicability evaluation for ALM scope understanding.

Several of classical project definition approaches though agree on to identify the project as a specific and unique endeavor to reach a set of goals with defined boundary conditions for example, scope, budget, timeline, quality (Kerzner, 2017; Wysocki, 2011a; Schwalbe, 2015) to even respecting projects as temporary organizations (Sydow and Windeler, 2020; Turner and Müller, 2003). These general project definitions are well known and widely accepted project definitions, which are fitting most of the projects in general product and project scopes. The further on it will be observable, that for ALM there are differences for example the scope definition for the ALM is more flexible than for regular projects, and for time scope definition is also more complicated, as for ALM can contain scattered development periods and even under the operation already running, and end of the acitivities coming not by the handover but rather by the end of lifetime of the ALM cycle.

In the definition by Wysocki (2011a), a project is a sequence of unique, complex, and connected activities that have one goal or purpose and that must be completed by a specific time, within budget, and according to specifications. By doing so Wysocki already highlighting the complexity, which is also part of ALM, and highlights the predefined scope definitions which are to be recorded for ALM also. However, this is not yet covering fully the conditions, for example the time factor for the projects is usually understood by the deployment time, but for the ALM this is not the case. Also, other factors like change management handling should be emphatically recorded too in the contracting phase already for ALM, which is open in this project definition.

ISO 21500:2021 (Guidance on Project Management) (Stefanova-Stoyanova and Danov, 2022) defined the project as a unique set of processes consisting of coordinated and controlled activities with start and finish date, undertaken to achieve an objective. For projects the general understanding for start time is the conception time is accepted, similarly for ALM also, however, for finish date for projects meant the handover to the customer after the development phase is ready, in the case of ALM the handover after the development phase initiates the deployment and starts the operations phase, which can also contain further development activities in the future. Thus the timeline for ALM is more extended compared to traditional project scope understanding.

World-leading expert project management organization PMI (Project Management Institute) in their PMBOK (Project Management Book of Knowledge), which is a standard in the practice-oriented world has by editions some small adaptations in the definition, however, the baseline is that a project is a temporary endeavor undertaken to create a unique product, service, or result (PMI, 2021). There are also scholars challenging the PMBOK for its insufficient adaptability, limited adaptability, and overhead for administration. The inflexible nature of PMBOK's procedures can be burdensome when applied to projects that require regular modifications or when dealing with shifting requirements. The proposed methods may have challenges in adjusting to swiftly evolving corporate settings and market situations. Administrative expenses can be raised due to the emphasis on paperwork and protocols, which can reduce efficiency for smaller projects (Gasik, 2015). ALM is relied on administrative environment also, mostly to be able to track the changes and evolution of the workproducts. So processes and tools are extremely important for ALM environment also. This is one of the reasons, the ALM tool vendors can have such an important role in shaping this area.

Westland (2007) claims that projects are different from standard business operational activities, as they:

1. Are *unique* in nature. They do not involve repetitive, identical processes.

2. Have a defined *timescale*. Projects have a clearly defined start and end date they need to deliver the content according to the requirements.

3. Have an approved *budget*, i.e. the level of financial expenditures fixed for deliverables produced to meet customer requirements scope defined in the contract.

4. Have limited *resources*. At the beginning of the project, an agreed amount of labor, material, and equipment are allocated.

5. Involve *risk*. Projects contain uncertainty and thus carry business risk.

There are modern project definition which are respecting the projects as change agents, as they are transforming an organization's operations, products, or services through a structured approach (Shenhar and Dvir, 2007). This approach is are least closest to the ALM approach also related to a project and service characteristics too, and also uses a structured approach, however, the more details for ALM for the SW-centric and interdisciplinary attributes not yet present here either.

After clarifying the understanding of the general project definition and ALM relation in a high-level overview, the next relevant question is examined related to the potential typology for ALM in the world of projects. As it was visible not clearly fit any project definitions, the expectation is also that no clear project typology matching is available for ALM either. However, as ALM is strongly oriented to new software application development and operations, thus also the typology for similar projects is to be researched based on its results. Therefore, based on result orientation rather than belonging to the Product Development Projects area Research and Development Projects area as both of them aiming to create of develop a new product, in this case, a software product (i.e. the application), or enhance an existing one. Also related to the R&D Projects as the ALM also shares the innovative characteristics, high uncertainty, extensive content and collaboration need, usually longer timelines, and significant investment in innovation (Stretton, 2021). The commercial Product Development related characteristics are fitting also well for ALM, as those are mostly market-driven, iterative developments, a strong focus on customer needs (Al-Saqqa et al., 2020). On the other hand, there are academics who are claiming that application software development is separated in typology from R&D and Product Development project as their goal definition is not clear, and therefore also the applied project management methodologies must be different (Kuchta and Skowron, 2016). So among the academics there are also dispute where to place the software projects related development, therefore for the scope of this thesis the Software development projects are more examined in the upcoming section to identify the relation with the ALM area.

The project typology can happen also related to the project stakeholders (Bahadorestani et al., 2019), which is an interesting aspect of ALM scope also. As ALM is strongly related to interdisciplinary work during its lifecycle, the interfacing of these competency areas demand a well usable solution between the stakeholders. ALM lifecycle is focused on creating a software application usually for an external customer, which means it is rather close to the client projects typization from this perspective. Even though the internal project types cannot be excluded either. So in overall, software application development projects are versatile and can fit into multiple typologies based on their goals, stakeholders, and the context in which they are developed.

The common point of the above descriptions is the determined scope of content and the fixed duration for the execution. It was shown how the general project definition is covering or lacking the scope and content descriptions in relation to the ALM approach.

2.1.2 Software Projects

As ALM is centered in software applications scope, within this section an introduction for software and application projects are presented for better understanding of the their scope.

In the Information Technology (IT) area, a *SW project* is typically defined as a planned and organized effort to develop, deliver, and maintain a software application or system, following a structured set of activities, processes, and methodologies. It involves using various software development technologies, tools, and techniques to achieve specific objectives, such as creating a new software application, enhancing an existing one, or resolving a software-related problem. Current software projects usually demand complex management involving scheduling, planning, and monitoring tasks (Alba and Chicano, 2007).

As the software development went multi-purpose in the recent decades, it is also important to highlight what is understood in software typology and for application.

Nowadays, Software Applications, or in short, Applications, are often used as synonyms for SW. However, they have distinct meanings. Software refers to a set of instructions, programs, or data used to operate computers and execute specific tasks. It encompasses a wide range of computer programs, including system software (e.g., operating systems) and application software (e.g., word processors, web browsers). In academic discussions, "software" is a broader term that encompasses all types of programs and data that enable the functioning of a computer system. Therefore already Chapin et al. (2001) and Forward and Lethbridge (2008) has initiated the typological summary for software, and based on their purpose, data orientation and other specificities created summary. Here in the listing a short typology is shown to understand what is included and not included into the application scope.
Software types based on purpose:

- System SW: Manages hardware and software resources (e.g., operating systems, device drivers).
- Application Software: Helps users perform specific tasks (e.g., word processors, database management systems)
- Embedded Software: Controls hardware devices and systems (e.g., firmware in appliances).
- Web Software: Runs on web servers and is accessed via browsers (e.g., web applications).

Software types based on deployment model:

- On-Premises Software: Installed and run on local servers or personal computers.

- Cloud-Based Software: Hosted on remote servers and accessed over the internet.
- Hybrid Software: Combines on-premises and cloud-based elements.

Software types based on development methodology:

- Agile Software: Developed iteratively and incrementally with a focus on collaboration and flexibility.
- Waterfall Software: Follows a linear and sequential approach to development.
- DevOps Software: Emphasizes collaboration between development and operations teams.
- Prototyping Software: Uses prototypes to refine requirements and design.

An application is a specific type of software designed to perform a particular function or set of related functions for end-users. Based on above listed typology Application software type based on purpose, and based on deployment model can by any of the listed above, and based on development methodology mostly agile or DevOps type. Applications are user-facing and serve specific purposes, such as word processing, web browsing, or graphic design. In academic contexts, "application" is a subset of software, specifically referring to programs developed to address user needs in various domains. For ALM this broader meaning of application is preferred.

Recently, IT project managers have been challenged to keep their projects focused and at the same time support their organization's need to adapt to changes and uncertainty in the business environment. For projects with a flexible project structure, scheduling includes deciding whether to implement specific optional activities and impose the related precedence constraints (Kellenbrink and Helber, 2015). To ensure the efficiency of the project organizations, flexibility is usually not desired in the late phases of projects.

2.1.3 Project Lifecycle

In project management, it is usual for a project to undergo a sequence of interdependent phases, each of which contributes to the overall *project lifecycle* (PLC). While the specific nomenclature and quantity of phases may vary across different contexts, these stages remain widely recognized and consistent in the literature. See Figure 2.1 about the Goal and Purpose correlation (Wysocki, 2011a) for PLC.

The concept of a project life cycle is understood that, during a given stage, certain requirements must be met or the project is not allowed to pass to the next stage. Despite the unique characteristics of projects conducted in different industries, many firms use a generic project lifecycle model. The most basic model often includes the initiation, planning, executing, closing stages (Kloppenborg et al., 2014). PMI also includes the monitoring stage after execution (PMI, 2021).

The Project Management Body of Knowledge (PMBOK) Guide, which is published by the Project Management Institute (PMI), presents many project lifecycle models to offer organization and a methodical approach to project management. The PMBOK defines the following main lifecycles(PMI, 2021):

1. Predictive (Waterfall) Lifecycle:

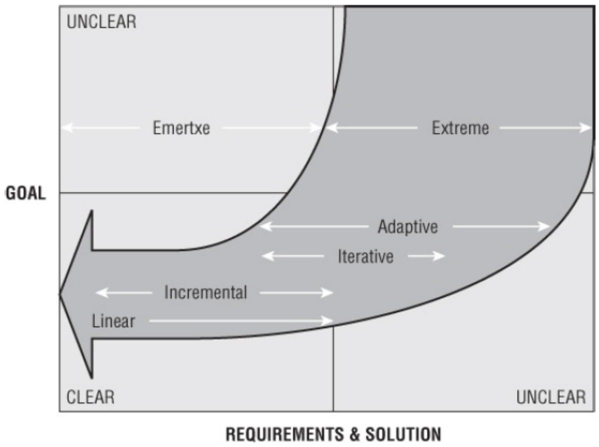


FIGURE 2.1: PLC approaches of Wysocki (2011a)

- **Description**: Also known as the Waterfall model, this lifecycle is characterized by a sequential process where each phase must be completed before the next one begins. It is plan-driven with defined scope, cost, and schedule.
- **Phases**: Initiation, Planning, Execution, Monitoring and Controlling, and Closing.
- **Usage**: Suitable for projects with well-defined requirements and low uncertainty.
- **ALM relation**: The execution of ALM has reoccurring and parallel activities and phases (Development and Operations), thus this approach is not appropriate to model will the ALM problem.

2. Iterative Lifecycle:

   - **Description**: This lifecycle involves developing a product through repeated cycles (iterations) and refining the product with each cycle. Each iteration builds on the previous one.
   - **Phases**: Planning, Iteration, Review, and Revision.
   - **Usage**: Suitable for projects where requirements are expected to evolve over time and require progressive refinement.
   - **ALM relation**: The delivery of SW Applications can be and usually are developed in an iterative process, so might partially fit for ALM also. Also fitting from the perspective of additional features implementations, however, the iteration's length for such models is fixed, at ALM this might not be ensured.

3. Incremental Lifecycle:

   - **Description**: An incremental lifecycle delivers the product through a series of increments or pieces. Each increment adds functional capability to the product.
   - **Phases**: Initial Planning, Incremental Development, Integration, and Final Deployment.

- **Usage**: Suitable for projects where delivering parts of the product early adds value to the stakeholders.
- **ALM relation**: The incremental approach for SW content development are common and usual, so from this perspective can be applicable also.

4. Adaptive (Agile) Lifecycle:

   - **Description**: Agile lifecycles are iterative and incremental but emphasize flexibility, customer collaboration, and responsiveness to change. They are adaptive to changing requirements and stakeholder needs.
   - **Phases**: Envision, Speculate, Explore, Adapt, and Close.
   - **Usage**: Suitable for projects with high uncertainty, rapidly changing requirements, or when frequent delivery of value is desired.
   - **ALM relation**: The Agile way of welcoming the changes fitting extremely well for the ALM understanding also, and usually in ALM agile is used by experience. The most fitting and most commonly used lifecycle currently from ALM is this Agile model.

5. Hybrid Lifecycle:

   - **Description**: This lifecycle combines elements from both predictive and adaptive lifecycles. Different parts of the project may follow different lifecycle approaches based on their specific requirements and uncertainty levels.
   - **Phases**: Vary depending on the combination of predictive, iterative, incremental, and adaptive approaches.
   - **Usage**: Suitable for complex projects where different components have varying degrees of stability and changeability.
   - **ALM relation**: Adapting Agile and other lifecycle models might fit well also for ALM, potentially in practice next to Agile such hybrid lifecycle model is used.

On the scope of attributes from the projects perspective, the ALM approaches might be similar to the above-described models.

These phases are often described as a stage gate model whereby a project must pass through an approval gate, by means of having someone approve a deliverable that was created during that stage, to be able to move from one stage to the next (Cooper, 2006).

In the IT area, including ALM, the life cycle models are mostly understood for the Software Project itself and contain specific steps. The Software Development Life Cycle, or SDLC, is a systematic process aimed at delivering software with optimal quality, minimum cost, and within a short timeframe. By providing well-defined stages, SDLC enables organizations to efficiently produce thoroughly tested software that is fully prepared for deployment. The development of quality software is achieved through a well-articulated SDLC model. The commonly successful SDLC models comprise Waterfall, Spiral, Incremental or Iterative, Rational Unified Process (RUP), Rapid Application Development (RAD), V, Agile, Synchronize and Stabilize, and Rapid Prototyping, among others (Akinsola et al., 2020). See a detailed comparison for SDLC models by Ragunath et al. (2010), Ruparelia (2010), Kute and Thorat (2014), and Akinsola et al. (2020).

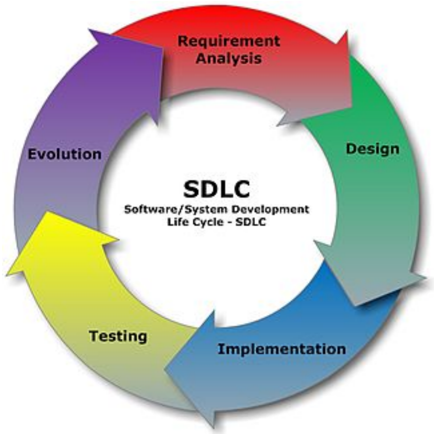


FIGURE 2.2: SDLC circle by ISO 12207
Source: ISO (2002)

Lack of the models though is that only some of them discuss the key issues like Change management, Incident management (Ruparelia, 2010) (for example the V-model for Waterfall or the Agile model (Akinsola et al., 2020) which are essential parts of ALM scope. ISO 12207 (ISO, 2002) depicts Systems and software engineering – Software lifecycle processes as seen in Figure 2.2 with the steps Requirement Analysis, Design, Implementation, Testing, and Evolution. The SW Life Cycle focuses solely on the SW development, testing, and deployment, and does not include the scope for maintenance and retirement of the application.

For the Application Lifecycle Management understanding, a detailed overview will follow in the upcoming Section 2.3, including the historical overview and relationship between the Product Lifecycle Management (PLM) and ALM, the evolution of the ALM scope with the main steps described and the significant role of tool vendors.

2.2 Project Management approaches

Modern project management emerged five decades ago out of construction projects. During the last decades though went through several changes and enhancements in scope and content also. The need for improvements and speed brought together with the computers and with computer-aided designs, later on, applied also for SW developments (Wysocki, 2010).

Westland (2007) claims that project management is the skills, tools, and management process required to complete the project, so it is rather an interdisciplinary activity to lead the projects to success. The Iron Triangle, also known as the Triple Constraint or the Project Management Triangle, is a core concept in measuring project success. It represents the fundamental criteria of delivering a project on time, within budget and meeting agreed-upon quality, performance, or scope standards. The Iron Triangle has become the standard for regularly evaluating project performance (Pinto, 2010). Even though several scholars are challenging these three factors, mostly agree that time and cost are mandatory parts, though, in the third corner, the quality, scope, or other factors are present. See a detailed review in the article of Pollack et al. (2018).

The understanding of the iron triangle (cost, time, scope) and the potential trade-offs between the conditions for project scheduling problems are crucial cornerstones in project management. Figure 2.3 demonstrates the side-wise potentials of how to manage the project cheaper, faster, or better (Van Wyngaard et al., 2012).

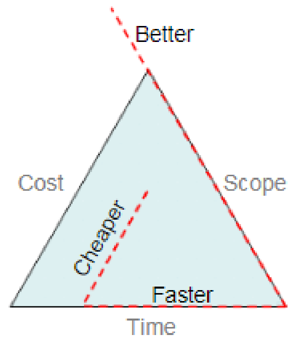


FIGURE 2.3: Iron triangle trade-off potentials.
Source: Van Wyngaard et al. (2012)

Selecting the proper project management approach is a crucial aspect of project handling (Charvat, 2003), as there is no single best method (Špundak, 2014). Based on the goal and solution clarity grouping of project management approaches defined by Wysocki (2011a), see Figure 2.4.

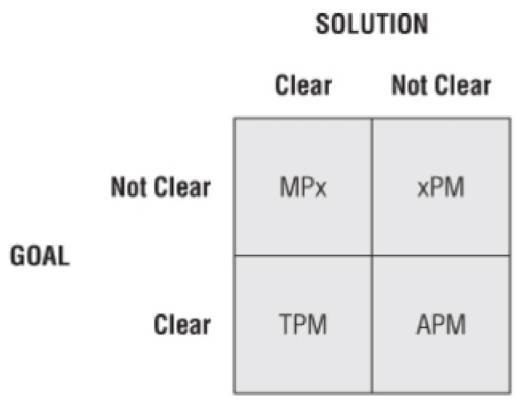


FIGURE 2.4: Project management approaches by Wysocki (2011a)

In case there is a clear goal and clear solution, it is called traditional project management (TPM). Construction projects can be good examples, where requirements are stable and no significant changes are expected.

In case there is a clear goal, but it is unclear the way to solve it, it is called agile project management (APM). Here the time and resources are fixed, and content can be changed within the iteration. Most SW development projects fall into this category.

Extreme project management (xPM) typically involves projects with ambiguous objectives and uncertain solutions, which is often the case in research and development or new product development initiatives. So in short it can be stated that xPM is a model appropriate for projects that have a goal in search of a solution (Wysocki, 2010).

On the other hand, the fourth category, emertxe (MPx), lacks a well-defined goal, but a solution already exists. So MPx is a model for projects that have a solution in search of a goal (Wysocki, 2010). This can occur when technology precedes its practical application.

It is also worth mentioning here the *projects managed with flexible approaches* for what it is paradoxical that while flexibility was frequently needed in the studied projects, they were rarely prepared for it, where flexibility is understood in the project planning and execution. As a consequence, structured approaches to project flexibility management are called for (Olsson, 2006a). Answering such structured need can be the multimode resource-constrained project scheduling in flexible projects (Kosztyan and I. Szalkai, 2020) where a matrix-based method provides scores for alternative project plans that host flexible task dependencies and undecided, supplementary task completion while also handling the new but unplanned tasks.

There are further project management approaches and extensions which are intentionally not covered here, as in this dissertation, the scheduling examinations focus will be on the TPM, APM and their combination, the HPM approach examination in the later phase due to their most frequent application in the software projects. Without claiming to be exhaustive, the following methods are also used in sw development and their further examination in a future phase of the research may be conceivable and desirable. Such methodologies and frameworks like Lean Project Management, Six Sigma, Critical Chain Project Management, Benefit Realization Management, Adaptive Project Framework, Integrated Project Delivery, ScrumBan or eXtreme Project Management. All of them either based on flexibility in handling or collaboration that are relevant in the ALM environemnt also, so future research are expected with them also. However, development methods which are strongly focusing on predefined content and fixed plans, like Projects IN Controller Environment are less likely to be suitable for ALM.

Applicability check for PM tools in the ALM environment has limitations as discussed at the beginning of the chapter already. Due to missing academic proposals for ALM environment, applicability tests with the given restrictions were conducted and presented in the following part of the dissertation for traditional, agile, and hybrid approaches and their application for the Simulation.

In traditional project management (TPM) approaches (such as construction projects or software development projects that follow a waterfall life cycle), the question is how much the realization of the requirements will cost. Therefore, while the scope is given and has to be completed, the time, cost, and quality are convertible if necessary. This approach allows more than one completion mode (technologies that require different time/cost/resource demands) (Creemers, 2015). In the agile project management (APM) approach, the question is how many of the features[1] can be included within the given budget and time interval (e.g., in a sprint). The overall goal is for all the approaches to realize the scope to the highest possible degree.

Following this brief introduction, let us now examine the approaches in greater detail.

2.2.1 Traditional Project Management

TPM is based on a well-worked out plan and its execution according to the processes. This linear view of the project from start to finish is also called a waterfall

[1] In the view of project management, to implement a feature is a task.

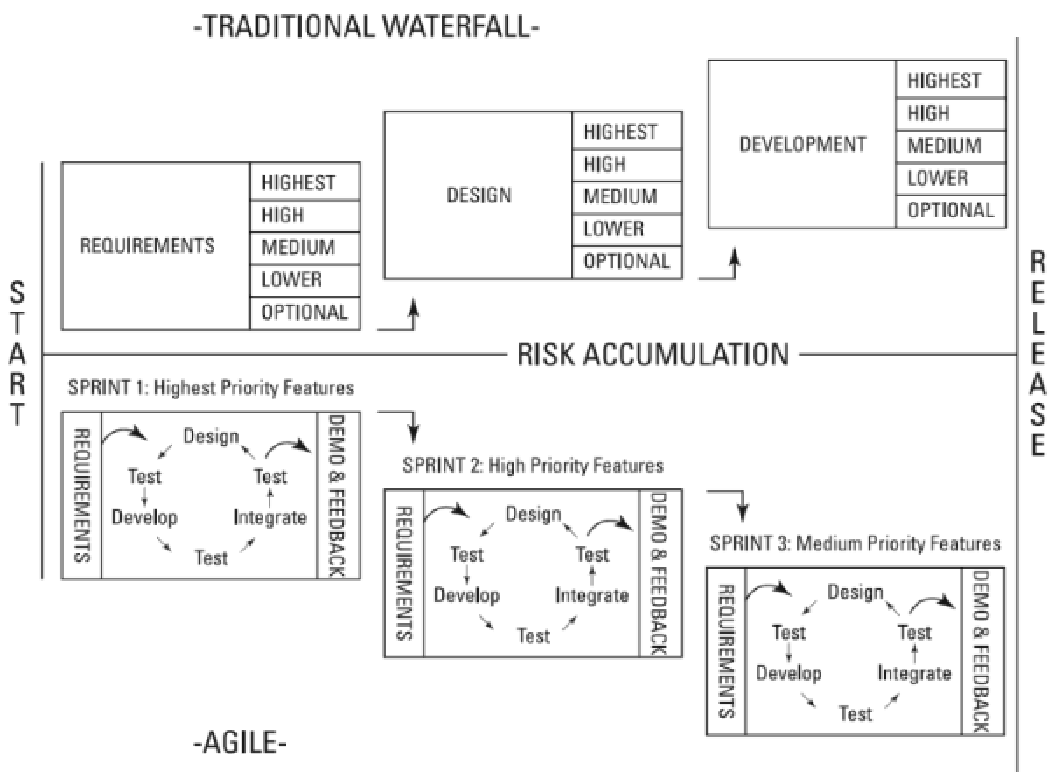


FIGURE 2.5: Traditional waterfall versus Agile structure.
Source: Layton et al. (2020) (Re-edited)

model in project management. The approach works successfully for simple projects with well-defined scopes or for those that have strong dependency, planning, and traceability with low uncertainties. The tasks follow each other like waterdrops in a waterfall, however, this structure is rigid, not reacting well to changes and turbulences. Worth mentioning, even though the traditional approach is highly based on a well-structured project plan, not every details can and will be planned here either. Approaches where all the tasks are planned to show into the standardization direction already, however, the project as the definition itself carries implicit something novelty. This is similarly the case for the SW applications development, very unlikely to use the exact same standardized scheme twice. Application of similar schemes though supporting quicker planning in traditional approaches also.

The traditional PM approach is widely used for software and application development due to its structured and well-defined sequential format it follows. In case the requirements are well-defined at the beginning of the development, and there is a relatively low chance for changes in them, there is a high chance that the software project can be successful. The CHAOS report is analyzing the waterfall or traditional approach since 1990's from several perspectives (Clancy, 1995).

However, recently the TPM got challenged in the VUCA world (volatility, uncertainty, complexity, and ambiguity), and the performance and success of projects handled this way are declining. Success factors become different in this new environment and therefore was the Agile approach as a suitable response to the VUCA challenges (Bundtzen and Hinrichs, 2021).

In Figure 2.5, it can be seen that the Agile iterative blocks are contrary to the traditional linear proceedings (Layton et al., 2020). The TPM approach is widely supported by traditional project scheduling methods see Brucker et al., 1999, for an excellent summary of traditional methods. Nevertheless, all of these methods are based on a fixed logic structure or a set of predefined alternatives (Servranckx and Vanhoucke, 2019; T. Servranckx and M. Vanhoucke, 2019).

2.2.2 Agile Project Management

The Agile Manifesto (see Figure 2.6) was created by the Agile Software Development Alliance (Fowler, Highsmith, et al., 2001) and since that time agile project management was applied and practiced in several areas beyond software development also. The co-existence with traditional project management and challenges in agile project management was analyzed (Ciric, Lalic, et al., 2018). Academics recognizing the demand for agile as PMI reported significant business growth based on its usage, thus creating a systematic literature review (SLR) about the challenges and their solutions in Agile project execution (Raharjo and Purwandari, 2020).

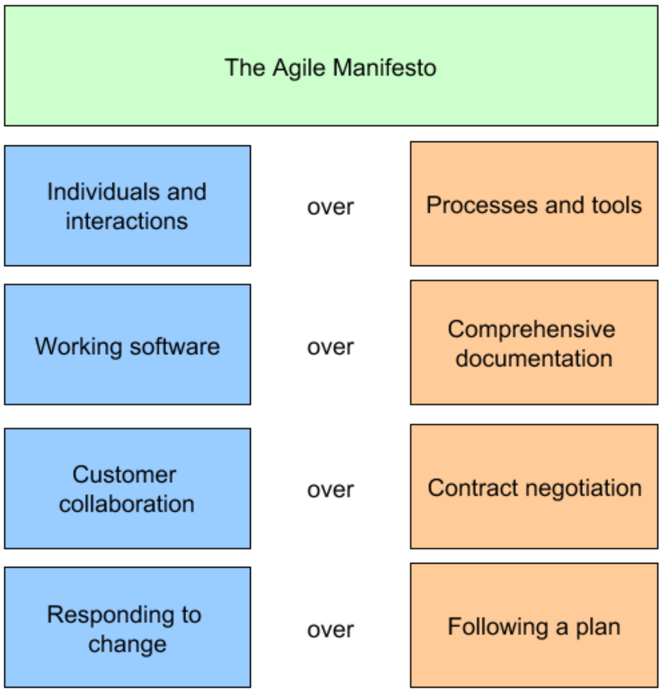


FIGURE 2.6: Agile Manifesto (Fowler, Highsmith, et al., 2001).

Some academics already announcing Agile as project management for the 21st century, Bergmann and Karwowski (2019) highlights that the Agile methodology in its early years and mainly focused on the SW has not impacted yet enough on project management. He created a review of the literature on agile and traditional in the project management domain and proposed project-type independent success factors. Gustavsson (2016) was also collecting benefits of Agile from non-software-related area applications. His ultimate finding is that the Agile Manifesto's first entity is the most universal advantage that often leads to downstream development pathologies. Serrador and Pinto (2015) was also surveying over a thousand non-IT projects looking for and confirming success improvements for agile projects based on efficiency and overall stakeholder satisfaction. He is highlighting Agile as a means to counter the dangers of traditional, front-end planning methods. Agile methods application facilitate collaboration and communication with iterative planning review, in contrast to the traditional method where a strict plan is followed.

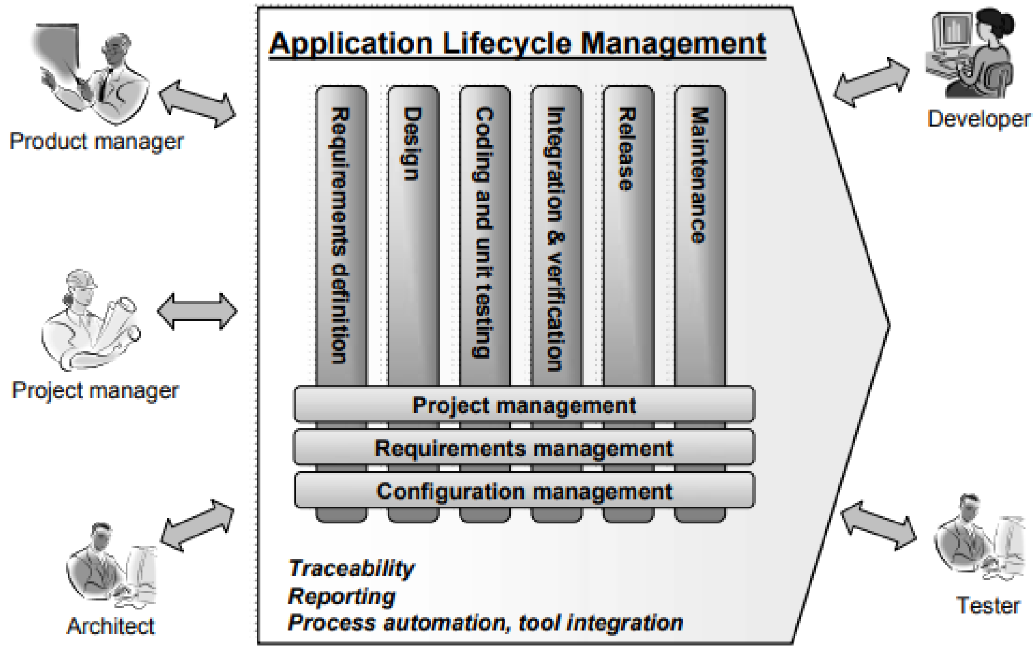


FIGURE 2.7: Application Lifecycle Management facilitates project co-operation and communication (Välimäki and Kääriäinen, 2008).

The agile approach also contains and shares the values with ALM as highlighted in Figure 2.7, where the actors continuously interact and proceed with the execution of the lifecycle activities.

In contrast to traditional techniques, the agile approach allows and sometimes requires restructuring the project. One of the main priorities of this method is to prioritize activities. Mandatory tasks have to be completed within a sprint (e.g., if the SCRUM method is followed) or within 2-3 sprints (e.g. if the KANBAN method is followed). Lower-priority activities can also be specified by other stakeholders. Nevertheless, if a sprint is specified and started, new tasks and new requirements can be implemented only in the next sprint.

2.2.3 Hybrid Project Management

Hybrid approaches are usually a mixture of agile and traditional project management approaches. See Table 3.2 for an overview of attribute comparisons. Hybrid project management is a flexible framework that is well-suited for the dynamic nature of software projects. This strategy utilizes the organized planning and precise documentation of conventional approaches to effectively handle certain project elements, such as meeting customer requirements and achieving important milestones. Simultaneously, it integrates Agile methodologies to improve adaptability to changing requirements, input from stakeholders, and iterative development phases. The hybrid model enables the prompt delivery of software increments, which facilitates continuous user involvement and iterative improvement. The ability to adapt is essential in software projects, as needs frequently change and technical breakthroughs quickly arise. Hybrid project management combines the rigor of predictive planning with the adaptability of Agile execution to provide effective risk management, optimal allocation of resources, and the production of high-quality deliverables. This dual strategy maximizes project results, addressing the intricate and diverse characteristics of modern software development environments.

An important distinction concerns agile projects that are embedded in traditional project plans (Theocharis et al., 2015), in contrast to agile and traditional approaches that are combined (Špundak, 2014) to manage single projects. Reiff and Schlegel (2022) She conducted a literature review to present a comprehensive analysis of various concepts and approaches related to hybrid project management, which encompasses both Waterfall and Agile methodologies. In addition, she has analyzed the benefits and drawbacks of the hybrid strategy, as well as its suitability and requirements. The effective application of the hybrid method relies on specific structural prerequisites that provide a more flexible project management strategy to address the constantly evolving needs and the unpredictable, highly complex, and volatile environment. In (Kosztyán and Szalkai, 2018), the authors explored the advantages and shortcomings of the combination of two worlds: agile and traditional project management approaches and techniques. However, as Pich et al. (2002) and Sommer et al. (2009) have previously stated, there is no superior project management approach. The choice of an adequate project management approach depends on the project's nature. Since all the traditional, agile, and hybrid project scheduling approaches can be implemented by computer algorithms (Kosztyán and Szalkai, 2020), the next step is to study which project management approach is the most suitable for different IT projects. While the current focus is on IT projects, the proposed simulation tool can also consider different kinds of other projects. Therefore, this model can be useful to estimate whether agile and hybrid approaches can be successful for other flexible but non-IT projects.

The nature of agile and hybrid projects, such as involving customers in the development process, ensuring strong executive support, and providing the ability to cope with emergent requirements, requires adaptive and flexible thinking for project management. In the agile project management (APM) approach, the completion of the project is more flexible, and the project structure can adapt to the changing customer requirements; see Figure 3.1. In all agile project management methods, project plans (i.e., backlogs) are split into smaller parts in order to be able to manage flexible agile projects. For example, one of the most popular APM methods, the SCRUM approach, suggests sprints that have to be completed within 2-5 weeks, while the other well-known method, KANBAN, restricts the number of work-in-progress activities (Dingsøyr et al., 2012).

So, in the overall summary of the three different management approaches which are the traditional, agile, and hybrid the main characteristics were introduced, explained the selection of them compared to other methodologies still proper for software projects. Starting up the examination of scheduling performance with these methodologies has high confidence to show already significantly different results.

2.3 Application Lifecycle Management

As seen above a recent approach to aid in the creation and management of work-products is known as Lifecycle Management. This method offers more efficient and systematic ways to support the development and management of complex products. Product Lifecycle Management (PLM) is the process of managing a company's products most effectively throughout their lifecycles. Application Lifecycle Management

(ALM), on the other hand, involves coordinating activities and managing artifacts (such as requirements, source code, test cases) during the lifecycle in the specific area, in the IT domain for software products or (software) applications. These concepts have primarily been developed and defined by tool vendors next to the academic community. This thesis focuses on ALM, particularly the development and post-development phases of the software lifecycle. There is a surprising lack of scientific efforts to define ALM and report practical experiences of deploying ALM solutions in an industrial setting. ALM solutions can be complex, incorporating various tools and practices for managing artifacts throughout the software development lifecycle, creating a need for supporting the development of such solutions in industrial contexts (Kääriäinen, 2011).

2.3.1 Product- and Application Lifecycle Management relation

The domain of Application Lifecycle Management is claimed to be a comprehensive software engineering approach that encompasses the entire lifespan of a software application from its initial concept, through development and deployment, to its ultimate retirement. ALM involves the management and coordination of processes, tools, and resources across various stages of the software development cycle, including requirements gathering, design, coding, testing, release, and maintenance. This high complexity of the combination of processes and artifact management, with apparent similarities though with distinctive differences of ALM to the PLM provides a challenge in the industry, as several practice-based articles and case studies are indicating from recent times (Deuter and Rizzo, 2016; Ebert, 2013; Deuter, Otte, et al., 2019; Duda et al., 2022).

Below, I present crucial components that contribute to the comprehension of the ALM domain. These factors serve to enhance knowledge and emphasize the notable disparities between the academic literature and the business-oriented understanding. This gap also highlights the need for more research and development in the ALM sector, including collaborative efforts between field professionals and the academic community.

Chappell et al. (2010) defined ALM as a continuous effort from three main aspects (Governance, Development and Operations), which are following the application lifecycle in time. See Figure 2.8 for details. The vertical lines depict three main phases: the Ideation, Deployment and End of Life. The horizontal line on the top represents Governance as a continuous feature across the complete lifecycle. Development below is scattered activity, which in the first phase has a more substantial duration, and post-deployment might reoccur with smaller-bigger entities based on the requests arriving. These can be scheduled or unplanned also, and from a scale standpoint vary from tasks up to subprojects. The re-occurring development phases describe the best specialty of ALM, compared to the general PLM approaches. The handling of such attributes faces a line of challenges for the traditional project management understanding for example for planning manpower, resources, cost, etc. The Operations line represents the early involvement already before the deployment, and the continuous supervision till End of Life.

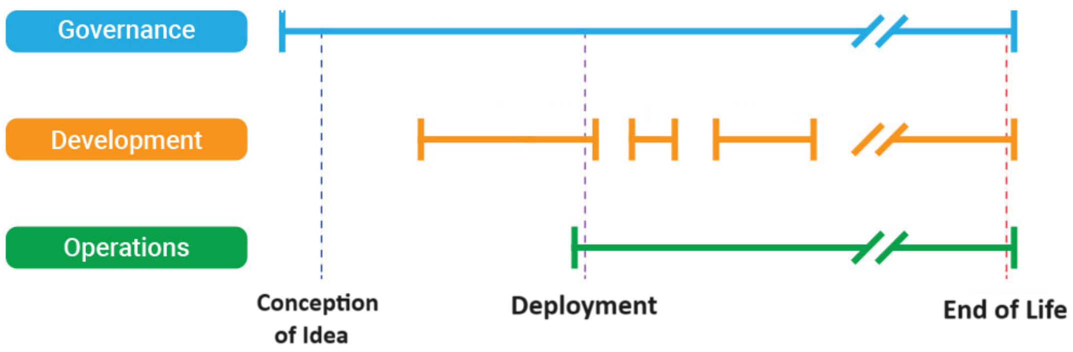


FIGURE 2.8: ALM process by Chappell et al. (2010)

Numerous academics and professionals concur that ALM also involves the management of related business processes, such as project management, quality assurance, and change management, to ensure the delivery of high-quality software products that meet user requirements and are delivered on-time and within budget. Overall, ALM provides a structured approach to software development that helps organizations to improve productivity, reduce risk, and deliver better software products (Kääriäinen, 2011; Rossberg, 2014; Rossberg, 2019).

Within the limited scarce literature, scholars also vary on how to define scope. Some authors draw strong parallels between ALM and PLM, asserting that ALM is only software-relevant PLM (Ebert, 2013; Deuter, Otte, et al., 2019). Others claim that ALM has a significantly wider scope (Kääriäinen, 2011). Therefore the main differences in understanding of the content and scope of ALM and PM are listed here for better understanding (Rossman, 2010):

- ALM is focused on the development and maintenance of software applications, while PM is mainly applied to the development of SW projects i.e. partial scope compared to ALM.
- ALM focuses on the software development cycle from start to finish, while Project Management may cover only a subset of the software development process or may cover non-software related projects.
- ALM is more technical in nature and requires a deeper understanding of software development processes and tools, while Project Management may involve a broader range of skills and knowledge.
- ALM is typically driven by the development team, while Project Management may involve stakeholders from multiple departments within an organization.
- ALM may involve more detailed and technical documentation than Project Management.
- ALM places a greater emphasis on software testing and quality assurance, while Project Management may not be as focused on these areas.
- ALM may require the use of specialized software development tools and technologies, while Project Management may use a broader range of tools and software applications.
- ALM may involve more frequent and smaller releases, while Project Management may focus on larger, less frequent releases.

- ALM may involve more iterative and incremental development processes, while Project Management may use more traditional waterfall or agile methodologies.

Organizations employ the SDLC and ALM procedures to create and manage software applications. There are, nevertheless, a few notable distinctions between the two. Here are the top five distinctions, as listed in Table 2.1 also.

Factor	**SDLC**	**ALM**
Scope	SW development only	Application from initiation till retirement
Integration	SW development only	Whole scale of PM area
Collaboration	Development Team	Dev. Team, Testing, Operations, Business
Automation	Only for SW code related	Development, Testing, Deployment, Release
Continuous Improvements	Rather a linear process, one time learning point	Feedback loops during the lifecycle

TABLE 2.1: SDLC and ALM differences summary table (Own edit)

For scope, SDLC focuses solely on software development, while ALM encompasses the entire lifecycle of an application, from development to retirement. ALM covers not only the development process but also the deployment, maintenance, and retirement phases of an application. Integration point of view ALM is a more integrated and comprehensive approach than SDLC. ALM encompasses processes such as requirements management, project management, testing, quality assurance, release management, and change management, while SDLC focuses only on development processes such as coding, testing, and deployment. The collaboration in SDLC is the focus of the SW development team only, however, for the ALM the collaboration of the connecting Testing, Operations (DevOps) and Business areas. Another key factor is automation, where SDLC is SW code-related only, while the ALM has automation for the complete chain, including development, testing, deployment, and release. This is based on the heavy agile approach applied during development. Finally, the Continuous improvements are in SDLC relatively simple and one-loop type, in the ALM there are several loops for learning.

Deuter and Rizzo (2016) is also highlighting that PLM due to its close routes to HW-related lifecycle reached its limit lacking the SW considered. Thus the ALM was introduced, to have the SW as the main consideration in the lifecycle development and management. In Figure 2.2 you can see the main characteristics of the PLM and ALM. Visible on the table, on the left side, the PLM contains the more generic product development-related activities, and ALM is more SW development and maintenance-oriented. Also, Deuter and Rizzo (2016) points out that the struggle for academics and business is already there due to the depicted significant differences.

Compatibility needs to be taken into consideration to confirm whether the PM tools are applicable. Although ALM and PM share many characteristics (Carmignani et al., 2017), such as development and implementation tasks, ALM places greater emphasis on maintenance, application lifecycle management, and the implementation of customer improvement requests (McNaughton et al., 2010). To assign a budget, the planning period as a timeframe might therefore be defined as

PLM IT Solutions	ALM IT Solutions
Traditional Project Management	Agile Project Management
Requirements Management	Release Management
Document Management	Requirements Management
CAx Integration	Document Management
Engineering Change Management	Integration of development Tools
Bills of Material Management	Source Code Management
Integration of Simulation Tools	Integration of Software build Processes
Workflow Support	Test Management
Problem Reports	Workflow Support
Product Configuration Management	Bug and Issue Tracking
Management of Product/Design Standards	Software Configuration Management
Integration to ERP systems	Management of Standard Libraries
Manufaturing Process Planning	Version Control
Materials Management	Task/Ticket Management

TABLE 2.2: Comparison of PLM and ALM core functionality
Source: Deuter, Otte, et al. (2019)

a set time span. In this instance, the resources are acknowledged as development (human) resources. When it comes to ALM, the substance of the scheduled activities (such as the launch of a new application) within the allotted time limit determines how points are calculated (Jakab and Novák, 2018).

Despite many commonalities, there is currently no widely accepted description or organizational model for ALM that can be used as a starting point for methodological research. In order to close this gap, a thorough examination of the literature was needed to define the ALM's parameters and offer a possible model for the application, which is one of the pillars of this dissertation, which can be seen in detail in the next chapter.

It is evident that systems and products have been more digitalized in recent years. Consider cellphones or modern automobiles with driver assistance systems — a growing number of items have some functionality supported by SW, or even have the primary functionality provided by a software product itself (Sinderen et al., 2006). The primary foundation for product management is the comprehensive implementation of Product Lifecycle Management (PLC) throughout the development process. PLC was essentially an improved version of Product Data Management (PDM), comprising a Bill of Material (BOM) and the associated project management

(PM) procedures. While several technologies had already been created to support physical goods, they lacked the functionality required for software creation. To close this gap, the Application Lifecycle Management (ALM) framework was also developed. Providing a thorough technological framework and solution for overseeing, managing, and controlling software development throughout the whole application lifecycle is the aim of application lifecycle management (ALM) (Deuter and Imort, 2021).

Mechanical, electrical/electronic (hardware), and software capabilities combine to form the overall functioning of smart devices. There are various lifespan models for software and hardware: whereas ALM concentrates on software, PLM concentrates on hardware. Manufacturers of smart products are compelled to gradually converge both lifespan models.

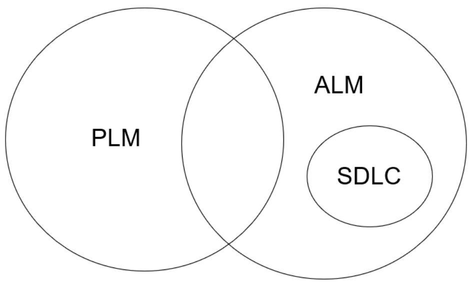


FIGURE 2.9: Relation among PLM, ALM, and SDLC (Own edit)

Despite this creative area's apparent importance, the research community usually leaves it to the PLM and ALM tool vendors, who are ultimately responsible for promoting the convergence (Deuter and Rizzo, 2016; Deuter, Otte, et al., 2019; Rao and Palaniappan, 2020). As Figure 2.9 shows the connection also for some of the PLM and ALM intersections for the project and program management elements, interlinks between HW and SW products, change management, collaboration and reports (Deuter and Rizzo, 2016). This shows also, that the tremendous amount of smart devices e.g., in ICT or even automotive, are facing the challenge for the ALM. In the case study, which is the third pillar of this dissertation, it will be shown also how the ALM is realized in an automotive supplier company.

From an organizational perspective, the ALM approach is prevalent in IT and SW development-related organizations. The main principle for adaptation is the structure follows strategy, which means that the ALM-related organization values appear in the organization structure also. Due to the frequency of adaptations, the SW-oriented work, mainly project-, matrix- or agile organizations are present where the ALM is partially or fully followed (Tüzün et al., 2019; Pirklbauer et al., 2009). This dissertation though not focus on the organization-related aspect of ALM, but rather on the technical and feasibility-related aspects.

In the following subsections, I will provide a brief introduction to the fields that have historically contributed to the ALM idea and discuss their specific contributions and limitations concerning the ALM concept.

2.3.2 Evolution of ALM comprehension over time

Service-oriented IT management is now standardized by ITIL. The British government ordered the Central Computer and Telecommunications Agency (CCTA), now the Office of Government Commerce (OCG), to optimize public administration with IT in the late 1980s. ITIL was born. ITIL best practices help IT organizations deliver high-quality, cost-effective IT services to clients. These initially complex and unstructured best practices have been significantly altered and adapted to changing contexts. Concerning lifecycle-oriented application management, what is most significant is that ITIL V3 is based on a service lifecycle approach that explicitly postulates the alignment of IT and business objectives as guiding maxims for the IT organization, and which in particular takes cognizance of the latest (IT) compliance rules (Arya et al., 2011a).

The first reference to ALM occurred in 2002 within the context of ITIL. The Office of Government Commerce in the United Kingdom regards ALM from a service management/operations perspective: ALM focuses on the activities that are related to the deployment, operation, support, and optimization of the software application. The primary goal is to guarantee that the application, once built and implemented, can fulfill the predefined service level (Hallerstede, 2013). Here ITIL focuses on itself the life of an application in a production environment. In the SDLC view, the development lifecycle starts with the decision to go ahead with a project, however, here it starts with deployment into the production environment. After deployment, the application is operated by the Operations responsible. Additional activities, such as bug fixes, and change management topics are handled by them.

Hallerstede (2013)'s opinion is that it is a prudent and beneficial perspective on ALM: Development and Operations are two components of Application Lifecycle Management (ALM), working together to oversee the full ALM process. It is essential to take into account both components from the outset when strategizing a development project; one cannot exist without the other. ALM differentiates between application creation and service management. Application development encompasses the ALM stages of requirements, design, and build, whereas service management encompasses the stages of deploy, operate, and optimize throughout a software's lifecycle. The application lifecycle commences with the collection of both functional and non-functional requirements. During the design process, these criteria are transformed into detailed specifications for the features. During the construction phase, the program and its architecture are implemented. New components are purchased or developed and later incorporated and tested. After the construction of the system is complete, the deployment phase commences. Hence, the modified architecture must be integrated into the current systems and the software must be made accessible. During the operational phase, it is necessary to provide assistance to users and effectively document any changes in the requirements. The last stage in the ALM cycle is the optimization phase. During this phase, the outcomes of operations are examined and evaluated. Hence, it is imperative to gather feedback from users and employ different methods of evaluation. The phases may not always occur consecutively, as they can overlap due to the presence of parallel circles, where several changes are implemented simultaneously, or iterations, where a new circle begins before the previous one is completed or when two or more process stages need to be repeated. ALM provides a comprehensive perspective on both the pre-launch (including requirements, design, and build) and post-launch (including deployment, operation, and optimization) stages (Hallerstede, 2013).

Marggi (2002) mentions that there is no clear definition existing yet, and the terms Application Management and Application Lifecycle Management were used as synonyms in the academic and business in 2002. Thus Oecking and Degenhardt (2011) go for the definition by breaking down the expression into 'application' and 'management'. Management is defined as the form and control of the problem-resolution process, which comprises the following aspects: planning, decision-making, assignment of tasks, and monitoring. On the other hand, man and machine constitute the subsystems of an information system whereas to be more precise, machines should be thought of as applications that can only run in a specific hardware environment. Application management forms part of this remit (Arya et al., 2011a).

Kaiser (2005) defines application management to be a comprehensive set of services provided by an external IT service provider that includes operating services for applications, project and implementation services, and ongoing development activities on a long-term basis. Typically, the contractual foundation for these services consists of set pricing components and service level agreements (SLAs). This introduces an important aspect of the financial and contractual aspects of the ALM scope. In this dissertation also the financial aspect is taken into account for the simulation and case study.

Kääriäinen (2011) in his summary expresses also scattered in the literature. The notion of ALM has primarily been explored in professional literature, such as Dooley et al. (2005), Doyle (2007), Schwaber et al. (2006) and Shaw (2007). The word ALM has often been superficially addressed or primarily discussed about ALM tools in numerous scientific studies, without delving into a comprehensive investigation of the ALM concept (see e.g., Dearle, 2007; Heindl et al., 2007; Moore et al., 2007; Medina-Dominguez et al., 2007). Weiss et al. (2009) and Göthe (2008) contend that the idea of ALM is ambiguous and that definitions are influenced by the marketing activities of tool suppliers. Rossberg (2008) asserts that individuals frequently conflate ALM with operations and maintenance, neglecting the inclusion of the development phase.

Numerous initiatives and conversations are now underway to achieve ALM and PLM integration. Nevertheless, the solutions mentioned earlier solely rely on PLM/ALM solutions provided by a single vendor. Despite the utilization of the Open Services for Lifecycle Collaboration (OSLC) standard in the case study, the solution remained exclusive to a particular vendor. Nevertheless, the process of designing and manufacturing digital products necessitates the integration of PLM and ALM technologies from many suppliers. To accomplish this objective nowadays, it is necessary to have custom interfaces (Deuter and Imort, 2021).

This is application management in the wider sense because it also includes application development services. Like Kaiser (2005), Marggi (2002) also bases his definition on the application lifecycle: Application management encompasses all controlling activities concerned with planning, building, and running an application. Marggi (2002) makes a distinction between this and application operation. This refers to the collection of the subservices of the overall operation that include operational activities for the operation of applications (Marggi, 2002). One criticism of the definition by Marggi (2002) is that it does not cover the entire lifecycle; the end of life of an application, its retirement, is simply ignored (Arya et al., 2011a).

For ALM also, it's essential to understand that all value creation is in scope on the business level highlighted by Rossberg (2019). This value is created by team effort, as the company personnel who play specific roles collaborate on projects to deliver business value to the organization. For the ALM the following, not limited, roles are key: Stakeholders, Business manager, Project manager, Product Owner, Scrum master, project management office (PMO), Business analyst, Architect, User experience (UX) design team, Database administrators, Developers, Testers, Operations and maintenance staff. These roles and their activities are adding up to the ALM process main part. Based on the organization, there are four distinguished views for ALM by Rossberg (2014) and Rossberg (2019):

- The software development lifecycle (SDLC) perspective is often used to understand application lifecycle management (ALM) as development has traditionally been responsible for managing the whole lifecycle of the program. This phenomenon may arise from the disparity between the business and IT departments within many firms, with IT assuming a dominant role.

- Service management or operations perspective: Regrettably, in experience, operations have been detached from IT development. As a consequence, Operations now possesses its distinct perspective on ALM, which has led to complications in this domain.

- Application Portfolio Management (APM) perspective: Due to the disconnect between business and IT, some firms have adopted a portfolio ALM strategy that encompasses IT development as just a minor component. From a commercial perspective, the emphasis has been on managing the portfolio rather than the full Asset Liability Management (ALM) process.

- Comprehensive perspective: Fortunately, several firms prioritize the holistic ALM process by including all three aforementioned perspectives. Adopting this approach is the sole method to get authority and enhance the efficiency of ALM. It is crucial for a Chief Information Officer (CIO) to maintain this perspective consistently, as failure to do so can lead to a loss of control.

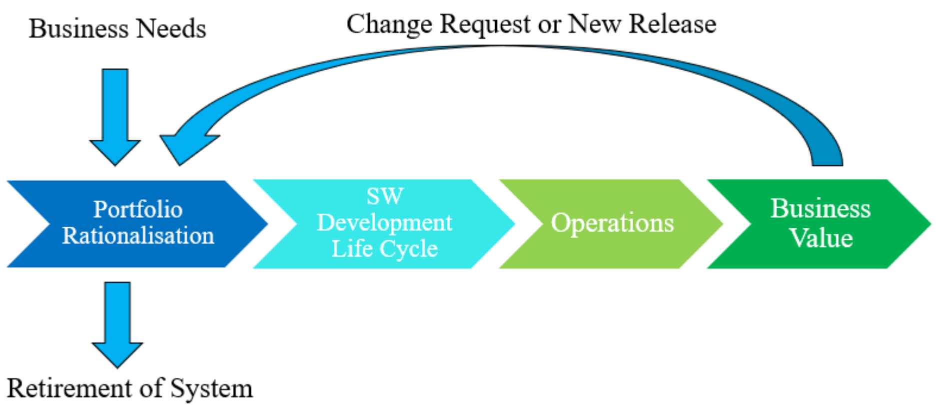


FIGURE 2.10: ALM process by Rossberg (2019) (Re-edited)

By this view, additional important aspects are getting into the scope of the ALM which is the portfolio level, and also the holistic view.

During the years an unavoidable evolution was happening in the definitions and understanding of the ALM scope, mainly driven by the vendors' and experts' needs.

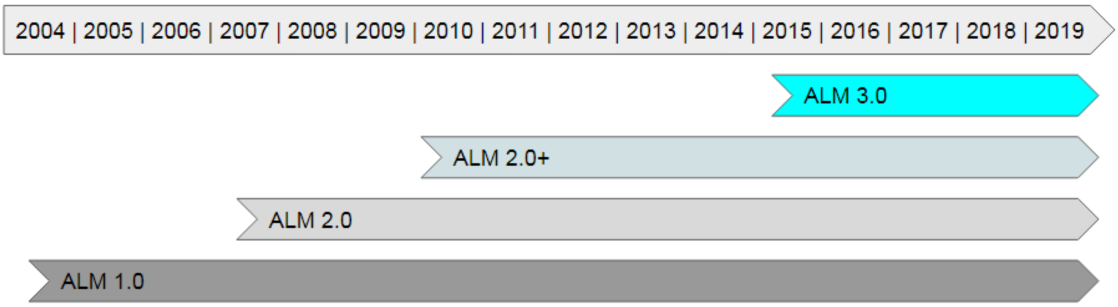


FIGURE 2.11: Generations over the years
(Source: own edit)

The first generation or version of ALM is referred to as ALM 1.0, with a distinct focus on each discipline. Proposing the utilization of diverse technologies and the integration of these tools as a way to solve the problem. The application lifecycle processes, including requirements management, design, development, build, and testing, utilize various tools for their management. These tools are integrated by exchanging information with other tools that manage different processes. The work of Aytekin et al. (2015) references the impact of Kääriäinen and Välimäki (2008) in their research. Even though for each discipline "best of breed" tools are selected, their point-to-point integration is definitively challenging, maintenance is expensive and specialized, improvements and features are limited, and the context switch time and cost might be significant overhead also.

The ALM 2.0 method provides solutions in the form of application software tools that have integrated processes. These tools are appropriate for the role-based approach to processes and the utilization of shared and integrated data. This technique facilitates inter-process integration by systematically providing communication and effectively ensuring multidisciplinary traceability. Simultaneously, effectively managing development processes can lead to speedier and more efficient creation of products. By enhancing efficiency, improved software quality can be achieved, expedited reporting, and accelerated delivery of process output contributions (Aytekin et al., 2015). The principle in ALM2.0 is then significantly changed, the toolchains are already created to support ALM purposes, focusing on feature development, and have a plug-in type approach, only what is needed can be taken from the full platform. Drawbacks are present here also, as the development of such a specific system is high in costs, the changeover from an existing system, therefore, can be expensive, and carries the all-in-one system's inbuilt technical and financial risks.

The ALM2.0+ incorporates enhancements to address the issues faced by the ALM2.0 platform. The integration of IT and ALM strategies in this context refers to the availability of a diverse range of solutions. This integration has the added benefit of facilitating cross-functional collaboration throughout ALM activities such as Work Planning, Traceability, Process Automation, and Reporting. The latest generation is called ALM3.0, whose principle is to have an efficient tool integration as a base so that the focus can be rather on the customer, and a learning organization (Rossberg, 2016).

So as visible, there are several approaches for defining the ALM content and context, however, there is not a clear and common understanding in the literature, that the evolution is ongoing and influenced by several factors. To support further theoretical research a more detailed approach is necessary to conclude. Therefore the indication for a systematic literature review is desirable to sustain a strong basis for scholarly works.

2.3.3 ALM tool vendors and their role

Application lifecycle management tools are a subset of the broader market for IT projects and product management. However, they specifically focus on establishing a more robust connection between business processes and software engineering. Organizations may streamline and optimize their application lifecycle management by selecting a software suite, resulting in time and cost savings.

In the market, PLM tools are already available and more mature. ALM tools are getting more focused, either using some modified PLM tools or developing specific ALM tools or tool sets. In some cases integration of the two tools is necessary, Brusa et al. (2018) are examining the integration of PLM, ALM, and PDM (Production Data Management) tools in his article.

In the 2010s the tool vendors were already leading the market and experts ahead of academia. Goth (2009) highlights that the demand for application lifecycle management (ALM) tools for agile development is rapidly increasing. Nevertheless, tool vendors and analysts are excessively preoccupied with competing for status to dedicate much time to fully acknowledging the irony of the situation rather than the seeming discrepancy with the Agile Manifesto. For instance, the self-contained principles of small agile teams may appear to be incompatible with the management needs of large enterprises. Nevertheless, other vendors employed the identical term—visibility—when characterizing one of the fundamental factors behind their products.

ALM tool features were collected and summarized in an academic article by De Simone et al. (2018), see Table 2.3, which shows the main features that an ALM tool on the market must contain.

ALM Feature	**Description**
F_1	Manage the lifecycle of work items and software artifacts via customized workflow
F_2	Store the artifacts in version control repositories, so every modification produces version history record
F_3	Enable real-time communication among actors by means of threaded discussions, wikis, notifications, alerts
F_4	Implement and assure the traceability links among the work items and SW artifacts involved into the process
F_5	Aid the collaborative work through concurrent access to all the work item and SW artifacts
F_6	Manage the roles, privileges, permissions of the actors in the process on the work items and SW artifacts
F_7	Monitor real-time the progress of the process exexution via customerized dashboards, reports and rich views
F_8	Enable comment on all work items, approve them, and veify approvals with digital signatures

TABLE 2.3: ALM tool features
Source: De Simone et al. (2018) (Re-edited)

The evolution and adaptation of the ALM tools in the market is happening at a fascinating pace, the competition is visible due to the competitive economic environment, see for comparison (StartupStash, 2022; Inflectra, 2024). However, in ovarall it can be stated that ALM tools assist in the achievement of several objectives for organizations:

- Enhance the quality of software by establishing a centralized repository for requirements, design, and test cases.
- Increase the effectiveness of software development by implementing automation to streamline procedures and establish a centralized repository of accurate information.
- Minimize the likelihood of errors by closely monitoring and documenting modifications made to both requirements and code.
- Enhance communication and collaboration among stakeholders by offering a shared workspace.

Ranking	ALM Tool Name	Tool Vendor
1	Spira Team	Inflectra
2	Azure DevOps	Microsoft
3	Rational Software	IBM
4	Rally (CA Agile Central)	Broadcom
5	JIRA+Confluence+Stash+Bamboo	Atalassian
6	Polarion ALM	Siemens
7	CodeBeamer	Intland Software
8	Kovair ALM Studio	Kovair
9	Digital.ai (formerly VersionOne)	Digital.ai
10	OpenText (formerly MicroFocus)	OpenText

TABLE 2.4: Top 10 ALM tool vendors and their tools from 2023 (Inflectra, 2024)

As there is a variety of ALM tools accessible on the market, each possessing distinct advantages and disadvantages. When selecting an Application Lifecycle Management (ALM) technology, thus it is possible to take into account specific requirements and expectation. The Table 2.4 displays the top ten items currently available on the market in 2023, rated according to broad characteristics. It should be noted that this list does not assert its comprehensiveness nor does an academic evaluation for them.

Even though ALM vendors including prominent historical companies from SW tools such as Microsoft, IBM, and HP which is shown in Table 2.4, are keeping a high focus on their products' most business value creation possible, there are some general inherent weaknesses by Regan et al. (2015):

- Traceability is primarily limited to the enclosed Application Lifecycle Management (ALM) system. Application Program Interfaces (APIs) exist for accessing internal data, but there was no defined open method of exchanging this data until the OSLC effort was introduced.

- Traceability reports can be generated to provide valuable information. However, these reports are static and do not reflect the dynamic nature of requirements and recognized problems, which can even come from sources outside the ALM system.

- The complex nature of the set of widgets, including buttons, text fields, tabs, and links, given for accessing and editing resource properties can easily confound assessors and users.

- Assessors and users must go through several links and tabs to access destinations, such as web pages and views. However, understanding these connections and tabs is not crucial for the assessment.

- Scheduling plannings are supported with limited automatization or manual plannings which need settings reviews regularly.

Moreira (2013) claims that regrettably, a comprehensive ALM solution that caters to all needs does not exist due to the extensive scope and complexity of full ALM, as well as the increasingly intricate and varied nature of software development. However, the greater the level of integration in a tool framework, the more an Agile Team can concentrate on creating client value. For proper tool selection Klespitz et al. (2016) was creating a recommendation for companies to select the proper ALM solutions fitting their purpose.

So as summary about ALM tools, it was shown that there are multiple solutions provided by several vendors support the lifecycle management in the development and maintenance phases. Their limited capabilities are bonded to the unclear content of ALM definition and are strongly vendor-driven. Further in this thesis, the tools are not detailed, the focus is on the further steps in this gap of ALM understanding clarification. A future collaboration between the tool vendors and the academics is possibly required e.g., to merge within the tool the scheduling algorithms developed by academia.

In overall summary of ALM as entity, after gaining a deeper understanding of the ALM background and its connection to IT areas, it is evident that the evolution of ALM is still in progress and not yet fully established. There remain numerous unanswered questions and opportunities for improvement in various aspects of ALM. It is evident that tool suppliers play a significant role in the development of the region. However, it is also apparent that their involvement is mostly driven by their commercial development goals rather than a focus on theoretical improvements. Adapting their tools and marketing them as a solution or solution kits for companies in need is a rewarding enterprise, as seen by the emergence of several spin-off companies alongside the established "big names" in the software development tool industry. The rivalry is an essential aspect of this period of evolutionary growth, when the principle of survival of the fittest may ultimately dominate. It is important to note that special software development needs have embraced the Agile Manifesto, which resulted to move away from the traditional waterfall-like method.

Therefore, influence of experts should not be underestimated in the IT industry. Although the academic society showed interest in the innovative setup offered by the ALM environment, no meaningful breakthrough has been accomplished or delivered yet. This might be attributed to many factors that are worth identifying, and discussed in the later part of this dissertation also. The focus of the dissertation is though mostly on the conceptualization and methodological aspects for academic work assistance. Hence, the forthcoming section of the literature will mostly concentrate on the processing and analysis with academic tools.

2.4 Systematic literature review for ALM definition

As shown in Chapter 1.1 Introduction, the ALM does not have a clear straightforward well-accepted definition, which is desirable for further theoretical and methodological research. Thus this section is proceeding with the presentation of the systematic literature review for covering this gap.

An important part of the academic background work is to establish the foundation of the investigated area, discover the breadth and depth of the existing body of work, and the validity and quality of the research materials. Such as identifying the scope, and the research materials availability. Even for the pre-screening of the area, it was visible that the narrowness of the area and the results showing only from recent decades will identify a limited and scarce base. Being able to proceed with the research, the first main step was to identify and research the area. Lacking an extensive overview and fulfilling literature study about the ALM area, the author decided to proceed with a literature review, which identifies the scope, i.e., the size of the research area, and the main characteristics of ALM definition by the scholars.

Even though the academic literature is seemingly quite limited, definitely worth mentioning that nonscientific articles (e.g., technical tool descriptions, business advertisements, training materials, ALM tool setup guidelines) are prevalent.

The preceding studies about the literature reviews are well summarized and analyzed by Paré et al. (2015a), who typologized the review types in their article for the information systems (IS) area. Systematic Literature Review (SLR) has the advantage of providing a comprehensive view with repeatable, rigorous methods. For proceeding with an SLR a detailed guidance is available that was followed (Xiao and Watson, 2019).

The evolution of consumer attitudes towards the utilization and anticipated functionalities of software products, specifically software applications, has undergone significant transformation in recent decades. The general trend shows that the application users demand their SW's high availability and regular upgrade of their functionalities. User experience is, therefore, a key factor for application developer companies (Yusof et al., 2021). This phenomenon challenged the application developers and vendors simultaneously. A changeover in thinking was necessary to support the frequently extending content, the continuously expected improvements, within shorter cycle times. This resulted in flexible project structures, extreme project handling methods, and agile development techniques coming alive mostly driven by business needs (Fitzgerald and Stol, 2017). Regarding the SW development life cycle already several methods have been elaborated, even an international standard has been created, the ISO 12-207, which is the standard that defines the software life cycle processes, and which can be adapted by any type of organization that is involved in the acquisition or development of software products and services

(ISO, 2002). Moreover, additional factors come into consideration, such as the imperative for continuous improvement and the necessity to remain up-to-date in the market, challenges that cannot be exclusively addressed through Software Lifecycle Management alone. However, Application Lifecycle Management (ALM) promises a multidisciplinary framework that can host this complex approach with systematic and quality-oriented solutions (Otibine et al., 2017).

Comprehending ALM requires being acquainted with the Product Lifecycle Management (PLM) concept. While the terminology may vary, PLM is mostly used for physical products and ALM for software, the underlying concept of managing the entire lifecycle of a product or application applies to both domains. PLM focuses on tracing and managing all the activities and flows of data and information during the physical product development process and also during the actions of maintenance and support to identify a new business model that integrates engineering processes and different tools. PLM strategy is to integrate all elements (people, processes, business systems, and information) that participate in product development, process, and support its lifecycle along the value chain (Garetti and Terzi, 2003). However, ALM contains SW-specific extensions over PLM, such as post-release non-planned activities for market demands, significant function extensions, or changes. An additional challenging component from the vendor site is the Global Software Development, where companies face a globalized setup for SW development with different timezones, socio-differences, and communication challenges (Chadli and Idri, 2017). A combination of these previously described challenges is the part of the environment where ALM is defined.

The primary targeted audience of this investigation is the academic community, to invite them to contribute and improve the theoretical and methodological repertoire of ALM. Secondly, the business decision-makers can find it interesting to realize the difference between conventional SW development and application development in ALM environment. This can help them to decide later on investing in the fitting and effective tools and methodologies as, during the phases of the SW development life cycle, several tools are used. For successful management of the SW development, the configuration and requirement management, development and test management, modeling and architect, issue and change tracking, reporting, and other tools must be also interlinked and traced (Kääriäinen, Eskeli, et al., 2009). Unfortunately connecting or integrating such tools is very often a challenging task. Fortunately, ALM can provide an ecosystem of integrated tools, processes, and domain technologies to ensure quality-driven application development (Carrillo and McKorkle, 2008). ALM integrates development, collaboration, communication, and knowledge management tasks and centralizes the management of users, projects and processes. Current ALM solutions either have a low-level multi-vendor integration realized with a basic versioning system that is not bringing the above-expected benefits, or there exist the expensive all-in-one single vendor solutions (Otibine et al., 2017). A new paradigm is appearing in the SW development, as a reaction to the frequent changes, that is called the agile method. It is gaining more and more space where rapid development is needed. In the Agile Manifesto, several traditional paradigms are challenged such as the scope of content priority, resource handling, and tool orientation (Beck et al., 2001). Due to the pressure for output in a shorter time, new methods are developed, like DevOps (Development and Operations), which aims to reduce the time between committing a system change and placing the change into normal production, while ensuring high quality (Ebert et al., 2016).

The concept of Application Lifecycle Management emerged in the last decades to fill the need for coordination of activities and to manage artifacts in the SW development projects. At first, it was realized by tool integration which is the root of ALM. Since 2006 several researchers have been trying to grasp the concept of ALM from their viewpoint, like Schwaber who claimed that companies are aware of the problem, but cannot handle it well (Schwaber et al., 2006). Doyle (2007) referred to ALM as a complex system development. Otibine et al. (2017) claims that ALM in a sense is a quality management tool, however, none of these definitions fully cover the ALM scope and content questions from a scholarly point of view.

Additionally, ALM tool vendors based on their business strategy and technical backgrounds dare to modify the definition and scope of ALM (Polit, 2004). Thereby numerous variations of definitions are available from business and academic sources and several changes can be observed that appeared in the understanding and content. Though in the IT area, there was recently a methodology review created by Pereira and Serrano (2020). However, the exact definition of different IT projects missing here too. Especially for Application Lifecycle Management, even though the area has already been researched for several years. ALM is a business-driven IT area, academic and business references are imbalanced and differ in ALM definition and understanding. Visibly, significantly more non-academic references are available today on the internet search engines. What is ALM? How is it defined? Otibine et al. (2017) recently also highlights that still, no clear definition exists. Multiple times there is only a short part of the descriptions in methodology-related articles, however, still lacks a systematic review of scholarly literature, therefore current **S**ystematic **L**iterature **R**esearch **Q**uestions (SLRQx) for this research aims:

SLRQ1: What definition exists for ALM in the academic literature?

SLRQ2: How is ALM defined, what are its main characteristics and scope?

SLRQ3: How can ALM definition be synthesized for methodological research?

Aligned with the Research Question 1 (**RQ1**), this study aims to identify or create an ALM definition that supports future methodological research. First of all, this systematic literature review method is for providing the underlying information to reveal the already existing definitions (SLRQ1), and also by a following critical review it can identify the main characteristics (SLRQ2), which can be used to create the defintion for ALM supporting methodological researches (SLRQ3, RQ1) see on Figure 2.12.

FIGURE 2.12: Systematic Literature Review Questions and Thesis's Research Question connections

Consequently, after a proper definition is available in academia, it gives a common base for methodological research, as currently lacking the proper identification of the scope. Thus for scheduling for example optimally applicable methodologies and methods also can be defined. Currently, businesses are using only the best-fit solution for their specific interests. This means due to the missing context, not optimal tools are used also. Such methods are based on best-fit with expected limitations, e.g., assuming a fixed logic plan, such as a fixed set of tasks and a fixed sequence of completion (Z. Kosztyán and I. Szalkai, 2018), however, for ALM unplanned tasks can appear also, that contradicts traditional project management planning. Eliminating this issue, Wysocki (2011b) claims that IT projects have Agile project management tools, however, no clear and strong base in the methodological area is available yet for optimizing. Z. Kosztyán and I. Szalkai (2018) propose a new approach, a matrix-based method using scores for alternative solution plans, which already contains unplanned tasks to take a step towards academic support of IT and ALM projects.

For the scope of the systematic literature review, as academic literature, including journals and conferences, are currently scarce sources of ALM thus the research is extended to high-quality peer-reviewed artifacts, such as published books, and academic materials, but omitted questionable quality level sources like business articles, webpages, patents, and standards as well. Expecting these sources to be based on scientific literature and using them to synthesize the information.

2.4.1 Applied Review Methods

Determining the breadth and depth of the research area the keyword-based systematic literature review (SLR) method was used to assess the extent and comprehensiveness of the study field due to its ability to provide a transparent, replicable, and comprehensive perspective. The review can effectively establish the criteria and restrictions, allowing for the identification, analysis, and interpretation of the relevant studies within this specific and limited field of knowledge. The Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) reporting guidelines were chosen to ensure a high-quality systematic literature review (SLR) (Page et al., 2021). Although the origins of the PRISMA declaration can be traced back to 2009 in the field of health science, its well-established structure and rigorous framework have led to its adoption in various other scientific disciplines, including IT, as a guiding principle for conducting systematic reviews (Damasceno et al., 2022). Following the completion of the ALM-related source selection, the critical review approach was used to identify and analyze the material. This method allows for a thorough examination of existing literature in search of the ALM definition and any potential alternative interpretations (Paré et al., 2015b).

This research included four primary stages for systematic literature review (SLR). Identification involves compiling a list of information sources and clearly defining the main objective of the systematic literature review (SLR). Next is the Screening stage, during which the studies are selected from the sources based on particular criteria and any non-relevant matches are eliminated. Next is the Eligibility stage, during which a comprehensive examination and complete reading of the text are conducted to determine the quality of the sources. Non-appropriate sources are identified and excluded. The Including stage involves utilizing the chosen sources to conduct a targeted evaluation, specifically for the critical review of the ALM definition review.

Discovering a newly developing and narrow area necessitates the use of specialized academic research techniques. Literature reviews are important for gaining a comprehensive understanding of a new field. This is because their rigorous techniques ensure that the systematic search yields comparable findings. Furthermore, it generates a compilation of existing knowledge. Nevertheless, employing a methodical strategy for reviewing is only somewhat effective in enhancing productivity due to the inflexible reliance on outdated technologies inside the academic publishing system (R. Watson, 2015). In Information Science (IS) there are several systematic literature review (SLR) typologies identified already based on their purpose used in top-ranked IS journals (Paré et al., 2015b). Though some academics argue SLR in IS is critically assessing their claims and implications (Boell and Cecez-Kecmanovic, 2015), such drawback of the SLRs is that they result in either a cross-sectional or longitudinal analysis. Present study the guidelines of the PRISMA method were used to make a systematic literature review for a cross-sectional analysis, that can serve as a base for further improvement for future longitudinal exploration. By employing this process, it is possible to conduct a systematic review of currently under-researched fields, leading to a comprehensive overview and a compilation of foundational publications.

2.4.2 Identification of sources

Prior to commencing the systematic investigation, a preliminary search was conducted on google.com for "application lifecycle management," yielding approximately 129,000 results, including various sponsored material from vendors and business-related information. This suggests that the sector is predominantly influenced by vendors, and there is a need for academics to enhance their understanding and conduct further study in order to model, create, and enhance approaches. Due to the strong influence of vendors, ALM relies heavily on information sources from both the public and corporate sectors. However, the reliability and quality of these sources cannot be guaranteed. A preliminary examination conducted in July 2023 using scientific literature databases such as Web of Science and Scopus yielded fewer results compared to Google Scholar. However, there were some similarities between the findings from these databases. Consequently, an evaluation was conducted to determine which source is the most suitable in terms of coverage and quality. Harzing and Alakangas (2015) has a comparison of the three main sources (WoS, Scopus, GS) from longitudinal and cross-disciplinary points of view. Halevi et al. (2017) created a review article that evaluated more than a decade also Google Scholar's advantages and challenges. Based on results from Halevi, stating that GS has a significantly higher amount and widespread results, also noting that in more than 60% of articles searched here by academics, therefore the decision was taken to utilize GS as the primary source for this research article to discover ALM. An exhausting comparison by Gusenbauer (2019) for the relevant findings among the main controlled databases, Google Scholar (GS) was found the widest, most comprehensive search engine covering concurrent database results also. However, Halevi et al. (2017) already drawing attention to using GS with caution due to the quality of resources indexed and overall policy. For explorative literature research though the author decided to use the widest source of information. Related to the estimated value for the GS results, it is confirmed that are only estimated in thousands and not a proper count (Sullivan, 2022). Handling this bias, for the filtered values already a direct count is proceeded in the later steps.

2.4.3 Search strategy in Screening

Following the adoption of GS, this section presents a detailed explanation of the search criteria, in accordance with the measures outlined in the PRISMA Statement.

Google Scholar is a search engine specifically designed for scientific literature. It offers advanced options, known as special search, which allow users to apply various filters. These filters include the ability to search for documents that include all specified words, specific expressions, or any words, or exclude certain search words. Users can also choose to search within the title of documents or within the full text. Additionally, Google Scholar allows users to search for documents by specific authors and within specific time periods. The objective is to incorporate peer-reviewed materials that are as close to the level of rigor as academic literature while excluding sources such as patents, standards, and non-academic literature. The issue is that these reviews if they exist at all, are primarily focused on professional and business aspects rather than undergoing a rigorous scientific peer review process. A drawback of using a keyword-based search is that if the naming conventions for the ALM are not adhered to, there is a risk of undiscovered information loss. This can be eradicated through a comprehensive examination of the pertinent literature papers.

The filter string used during the prescreening was "application lifecycle management". If all the terms are present in the finding, the GS setting can be used. There are no anticipated restrictions. The objective of this search is to display the broadest possible range of ALM articles, yielding a total of 1,470,000 results. Upon studying the entries, it became apparent that several of them did not include the specific phrase, but rather had variations of the familiar terms "product lifecycle management" and "application," among others.

The standardized PRISMA flowchart, depicted in Figure 2.13, outlines the essential four primary stages, which are indicated on the left as consecutive steps: Identification, Screening, Eligibility, and Included stages. Each stage is accompanied by filtering criteria. On the right-hand side, the excluded entries are displayed at each step.

In the Identification stage, the systematic review involves identifying the basis sources from which the entries are selected. During the preliminary prescreening in the **Identification** phase, it became evident that further filtering was necessary. As shown in Figure 2.13 in the Identification stage, the specific term "application lifecycle management" was utilized, resulting in an estimated 3,230 Google Scholar hits. From a chronological perspective, the initial occurrence was in 2000, as opposed to 2003 and 2005 for earlier publications on prescreening.

During the **Screening phase**, supplementary filters were implemented. Filter 1 utilized a keyword search to precisely locate the complete phrase "application lifecycle management" plus the term "definition". The search was conducted, taking into account the title, abstract, and content within the period frame of 2000-2023, focusing on the temporal domain. The filter was configured to include just scientific content, without any patents or references/quotes. The anticipated number of hits was reduced to 2,510, after deleting 720 records from prior searches. The primary emphasis in Filter 2 was on language. The database only included documents written in English, as it appeared to be the predominant language in the search results. Only a limited number of German, Turkish, Hungarian, Chinese, Korean, etc. entries were detected, falling below a considerable threshold. Practically, the GS findings are incorporated into the author's personal GS library (indicated by stars in the result section) and subsequently exported as a .csv (comma-separated values) text file. In the R Studio program, which is an integrated development environment used for statistical problem solutions and representation. The process of importing involves using a script to construct a database. This database was then used to identify and remove any duplicate items. The resulting data was saved in a spreadsheet,

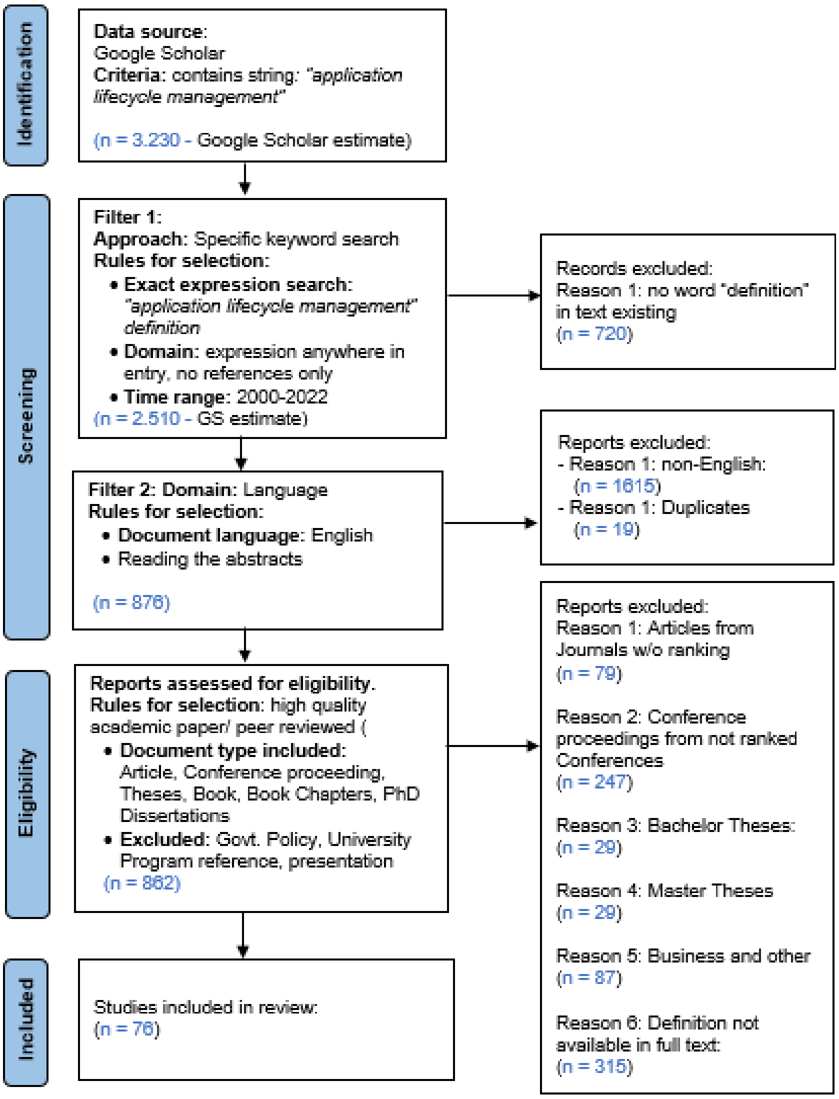


FIGURE 2.13: ALM definition search process depicted with PRISMA flowchart

which was necessary for the subsequent steps. There were a total of 876 records that remained after applying this filter.

2.4.4 Eligibility process

The eligibility criteria were carefully tailored to ensure an accurate selection of the appropriate document types with high quality. Given that GS includes nonacademic sources in its index, it is imperative to address the quality level of the documents. It is crucial to maintain high-quality information sources, so a scoping review (Pham et al., 2014; Kircaburun et al., 2021) was conducted to identify the types of academic and non-academic sources.

Given the substantial volume of input in articles, conference proceedings, books, and book chapters, it was deemed necessary to make further refinement.

Journal ranking is used to assess the quality of articles published in a journal. Subsequently, all the entries were examined to determine the journal in which they were published. The journal rating was then verified using the SCImago Journal & Country Rankings (SLR) website (www.scimagojr.com). While there may be some controversy surrounding the use of SJR in academia (Mañana-Rodríguez, 2015), there are no specific restrictions that would prevent doing this research. The SLR method identifies quartiles ranging from Q1 (highest) to Q4 (lowest) and assigns them based on the year of publication. If the journal does not have a ranking, it is classified as a non-ranked source. Refer to Table 2.5 for a summary of articles ranked according to the hosting journal's ranking in the year of publication. The initial column displays the ranking according to Scimago quartiles, ranging from Q1 to Q4, as well as the non-ranked entries.

TABLE 2.5: Articles ranking based on their Journals

Journal rank	Number of articles
Q1	25
Q2	27
Q3	21
Q4	14
Not ranked	79
Total	166

A total of 166 articles were evaluated as prospective material. Out of the total number of journal papers, 79 were from non-ranked journals, accounting for over half. This suggests that approximately half of the findings may not meet the required quality standards. Among the articles in the ranked journal, the highest number of occurrences were found in the Q2 level, with a total of 27 entries. This was followed by the Q1 level, which had 25 entries. Q4 had the fewest number of entries, with only 14 objects detected. Overall, the upper two quartiles have a little higher representation than the lower two quartiles. The year of publication is taken into account, as rankings might undergo considerable changes on an annual basis. Subsequently, this information was documented in the spreadsheet as well.

Determining the ranking of conference proceedings there are two primary rankings, first one is the Excellence in Research in Australia (ERA) which utilizes a three-level grading system. The grades are A (the highest), B, and C (the lowest). All conferences from the sources were thoroughly examined and selected based on their relevance to the topic. Refer to the screening results in Part (a) of Table 2.6, where the first column displays the ERA ranking levels and the second column shows the number of proceedings that were detected in the search.

The second ranking is the Qualis rating for conferences, which utilizes the H-index as a metric to evaluate the performance of conferences. The conferences are categorized into performance classes based on their H-index percentiles. These classes range from A1 (=best), A2, B1, ..., B5 (=worst), with a total of 7 levels. All the entries were verified according to Qualis, and the documented results can be found in portion (b) of Table 2.6.

TABLE 2.6: Conference proceeding rankings

(A) ERA ranking for Conference Proceedings

ERA Rank	Number of Proceedings
A	27
B	41
C	42
No ranking	271
Total	381

(B) Qualis ranking for Conference proceedings

Qualis Rank	Number of Proceedings
A1	17
A2	14
B1	26
B2	33
B3	16
B4	23
B5	5
No ranking	247
Total	381

For ERA ranking, from the total 381 Proceedings identified, 271 were presented in the non-ERA ranked conference, which means 71% of the proceedings are most probably not good enough quality level. Merely 7% was in the top, i.e. ERA A ranking with 27 entries. For ERA B and C, both are around 11% of the total amount, indicating that they represent the middle and bottom region of the quality based on ERA ranking.

For Qualis, Out of the entire 381 conference proceedings here in the non-ranked 247 entries, approximately 65% were only present. The remaining 7 levels encompass the remaining 35% but with more granularity than the ERA classification. Levels A1 and A2 account for 8%, and levels B1-B5 the 27% of the total entries. The majority of the submissions in Qualis are from the intermediate division. The B2 level contains 33 entries, followed by 26 entries at the B1 level, and 23 entries at the B4 level. The lowest level is located on B5 and has only 5 entrances. The frequency of high-quality level conferences is minimal, while conference proceedings are predominantly found in the B-level conferences.

When conducting research, it is important to carefully consider the sources used, both academic and non-academic, in order to establish a clear standard of quality for inclusion and ensure proper evaluation for acceptance (Kircaburun et al., 2021). In order to maintain a broad breadth while ensuring high quality, the article accepted scholarly sources with peer review, as indicated by experts' recommendations (Xiao and Watson, 2019). Prior to selecting criteria, a preliminary examination of the sources was conducted to assess how the acquired information aligns with the qualifying criteria, with a focus on utilizing peer-reviewed resources of superior quality. Out of the total Article submissions, 87 were classified as Q1-Q4, which accounts for about half of the entries, while the remaining 79 articles had no rank assigned to them. Unranked journals cannot be assessed for their quality and whether they underwent peer review. Therefore, these journals were omitted at this stage.

The ERA ranking for the Conference proceedings yielded a ranking for fewer than 30% of the total findings, which is relatively low and limited to only 3 category levels. The Qualis rankings had greater coverage and consisted of a total of 7 category levels, allowing for more precise differentiation. The choice was made to exclusively adhere to the Qualis rating in this study in order to ensure comprehensive coverage.

A summary of non-journal published academic resources can be found in Table 2.7, namely in portion (a). The first column contains the names of the types, while the second column displays the corresponding number of identified entries. The Theses were categorized into three distinct types: Bachelor (BSc), Master (MSc), and PhD Theses. The books and book chapters were segregated due to the presence of multiple distinct chapters in the entries. The outcomes of the prescreening process for business articles and materials can be observed in Table 2.7, namely in portion (b) of the table's first column. Business-related articles and published materials mostly focus on promoting and generating income for a specific product or service. Various types of papers are associated with technical and business case feasibility studies.

TABLE 2.7: Other type of sources occurance

(A) Theses and Books entries

Theses and Books	Number of entries
Bachelor Theses/	29
Master Theses	29
PhD Dissertation	7
Book	40
Book Chapter	123
Total	228

(B) Other categories

Other Categories	Number of entries
Business	60
White Paper	17
Technical Paper	5
Working Paper	1
Conference Poster	4
Total	86

In part (a) of Table 2.7, a total of 40 books were released during the analyzed time period. The biggest number of book chapters, specifically 123, addressed ALM. This could potentially be connected to the previously disclosed proceedings. A total of 29 submissions were identified in both the BSc and MSc screenings, while the PhD screening yielded the lowest number of entries, with only 7.

In part (b) of Table 2.7, the largest number of entries, specifically 60, were discovered in the Business category. There are only 1 working paper, 5 technical papers, and 17 whitepapers remaining. Whitepapers are considered authoritative and sophisticated documents that provide in-depth analysis and insights. As a result, there is a growing need to thoroughly examine the difficulties within a specific industry.

Targeting the peer-reviewed academic quality level, The Ph.D. dissertations were evaluated for their adherence to the rigorous standards of peer-reviewed academic quality. These dissertations underwent a thorough review by scholars during the doctoral process and required approval from the universities' doctoral board. This approval is essential for achieving the academic standard necessary to obtain a doctoral degree and have the dissertation accepted. The review quality of Bachelor and Master theses cannot be guaranteed to meet scholarly standards, as some colleges allow external experts without doctoral degrees to serve as opponents. Due to the inability to guarantee academic excellence, it was decided to remove BSc and MSc theses. The reviewing process is an integral component of publishing books and book chapters, so these entries are also included.

The business-oriented papers were eliminated because of the lack of clear peer reviews and reliability of sources, which seemed to prioritize commercial interests rather than academic rigor. The remaining papers, including white papers, technical papers, and working papers, already offer a formalized and structured perspective on the subject. Some of these papers even undergo peer review, although the extent of this review cannot be guaranteed. While the content of these papers is more advanced than that of a business source, it still falls within the realm of non-academic peer review. Therefore, they will not be considered for the next steps (Okon et al., 2020). Conference posters, due to their concise nature, are not suitable as a foundation for a critical evaluation.

After the Screening procedure, the items that were passed were next subjected to the Eligibility check, as shown in Figure 2.14. To summarize, only scholarly sources are recommended, and the figure illustrates the level of involvement for each segment of the pie chart.

The screening process encompassed a total of 391 items, which consisted of scholarly articles from reputable peer-reviewed journals, published proceedings from esteemed conferences, PhD theses, books, and book chapters. Excluded from the total of 471 entries are various types of publications, such as those from non-ranked journals, non-ranked conference proceedings, business-related papers, white papers, technical papers, government policies, university syllabus, presentations, and posters. The pie chart in Figure 2.14 displays the distribution of categories and their ratios in relation to the overall findings.

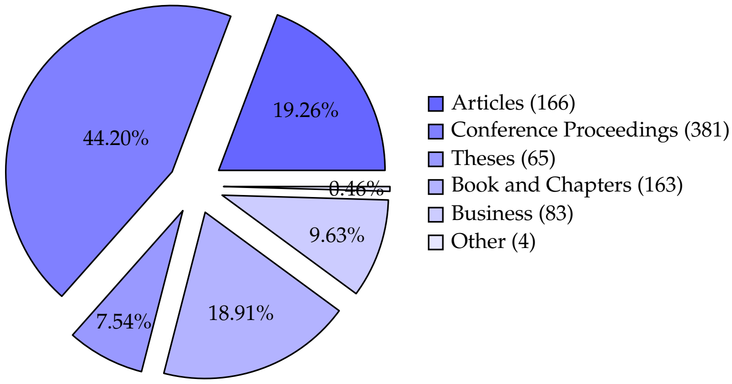


FIGURE 2.14: Typology distribution after scope screening

The majority of the entries, over 44%, consist of conference papers, suggesting that experts are already engaged in discussions on ALM subjects. Articles contribute the second highest number of entries, accounting for almost 19%. This indicates that a substantial and considerable number of items come from peer-reviewed academic works. The books and book chapters constitute the third largest category, accounting for around 19% of the total. This suggests that the ALM field provides support not only for academic work but also for professional activities.

As above mentioned the publication year of the screened entries was also recorded, in Figure 2.15 the yearly distribution can be seen for the Articles (blue), Conference proceedings (red), Dissertations (green), Books (purple) and Book Chapters (turquoise).

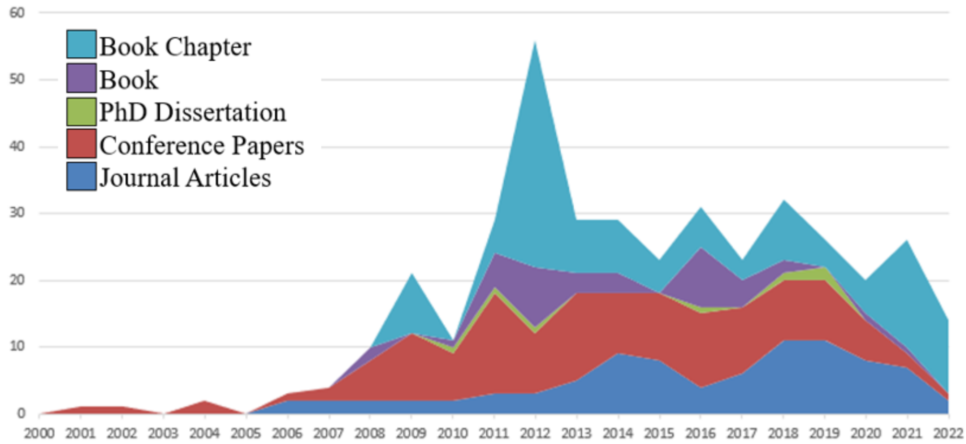


FIGURE 2.15: Articles, Conference papers, Dissertations, Books and Chapters yearly distribution over the years after the Screening

Figure 2.15 illustrates that the ALM had its initial conference entries in the 2000s, albeit in limited quantities. The conference entries and scientific publications on the issue began to increase only after 2005. The peak occurred in 2009, with the majority of book chapters being related to the conference papers published at the same time. The journal papers experienced a notable surge in 2014, followed by another rise in 2019, with approximately 10 entries. This indicates that there is already some existing scholarly work on the issue, albeit in a restricted manner. The conference proceedings from 2008 to 2019 demonstrate a consistent level of participation, with entries typically in the double digits. However, starting in 2019, there is a noticeable and significant decrease. The Dissertations are only available in a limited quantity, namely in the color green. Additionally, only a handful of them was produced, and some of them are associated with publications. Books, characterized by their purple color, are primarily associated with ALM, first emerging in the 2011s and reaching their peak in 2016. The book chapters experienced a modest surge in 2009, which can be attributed to the substantial rise in conference entries. Additionally, there was a big climb in 2012 following a delay in the publication of conference entries, along with the release of numerous books. There is a clear trend of decreasing numbers of publications, conferences, books, and chapters, suggesting that the past decade has seen a reduction in academic content related to ALM. However, it is evident that new energy and progress are required in this field.

The top ranking consists of 52 entries, which are the combined total of Q1 and Q2. On the other hand, the bottom 39 entries are represented by Q3 and Q4. In the Qualis rating, the A1, A2, and B1 categories reflect the highest level of quality, and so they are allocated to the top ranking. In summary, the set of materials labeled as "Top Academic" consists of articles from highly ranked journals and conference proceedings.

TABLE 2.8: Selected materials grouping

(A) Top Academic ranking distribution

Group content	#
Article from Journals Q1	25
Article from Journals Q2	27
Proceedings from Qualis A1	17
Proceedings from Qualis A2	14
Proceedings from Qualis B1	26
Total	109

(B) Extended ranking distribution

Group content	#
Articles from Journals Q3	21
Articles from Journals Q4	14
Proceedings from Qualis B2	33
Proceedings from Qualis B3	16
Proceedings from Qualis B4	23
Proceedings from Qualis B5	5
PhD Dissertation	7
Book	40
Book Chapter	123
Total	282

Table 2.8 section (b) shows the compilation of Extended Academic materials, including the remaining lower ranked journals Q3&Q4 (35), conference proceedings from B2-B5 (77), and PhD Dissertations (7), and a separate set of Books and Book chapters. In this Extended ranking group, the quantity of publications is lower, while the quantity of proceedings is higher in comparison. The majority of entries in the Extended Academic ranking are contributed by the Chapters, which are derived from individual Books and Proceeding chapters.

These two sets are designed for the purpose of conducting a source quality-based analysis of the findings in subsequent stages.

An additional expansion may have been pursued to augment the quantity of high-quality sources, in the event that a retrospective examination of the cited sources from the top-ranked academic sources category is also conducted. However, it is important to note that this work does not include any longitudinal research.

During the final stage of the Eligibility process, a full-text reading of the remaining 391 sources was conducted. The objective was to examine the filtered and screened documents in order to identify any definition that explicitly pertains to ALM. An additional 313 papers were removed due to the absence of a specific definition, resulting in a total of 78 sources that were included. Please refer to the last section of Figure 2.13 for further details.

2.4.5 Included sources

After implementing the aforementioned procedures, a total of 78 pertinent sources remained in the collection. Figures 2.16 and 2.17 illustrate that only a small number of items addressed the definition well or cited an academic source for a definition.

The academic field with the highest rating is visible in Figure 2.16. The blue columns reflect the articles from Journal Q1 and Q2, as well as the conference proceedings with Qualis ranks A1, A2, and B1 after the prescreening process. The red columns adjacent to them indicate the sources where the ALM definition is present and are included in the final stage of the PRISMA process. The selection method resulted in a total of 109 sources in this top academic sources, out of which only 20 included the definition of ALM.

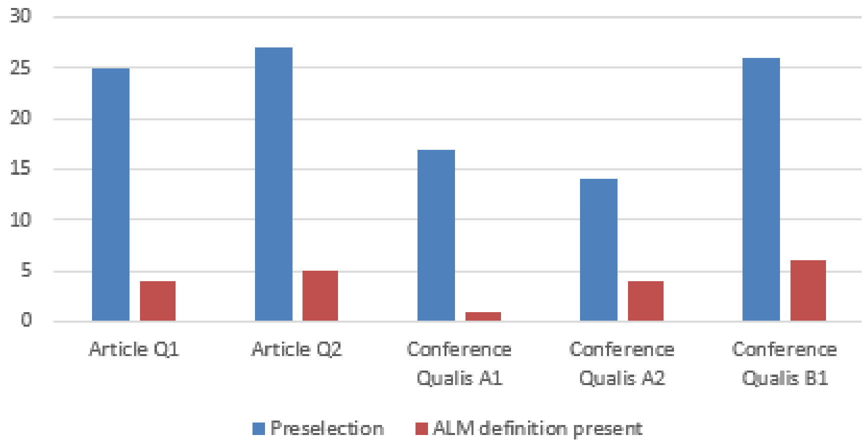


FIGURE 2.16: ALM definition present in preselected top-ranking academic sources

Figure 2.17 illustrates the extended academic sources. The blue columns represent the number of sources obtained after screening articles published in Q3 and Q4 journals, conference proceedings in Qualis B2, B3, B4, and B5, PhD dissertations, books, and book chapters. Among the 282 academic sources examined, only 56 entries contained the concept of ALM. These 20 sources from the highest-ranking academic sources and the 56 sources from the extended academic entry will form the foundation for the critical examination of the ALM definition in critical research.

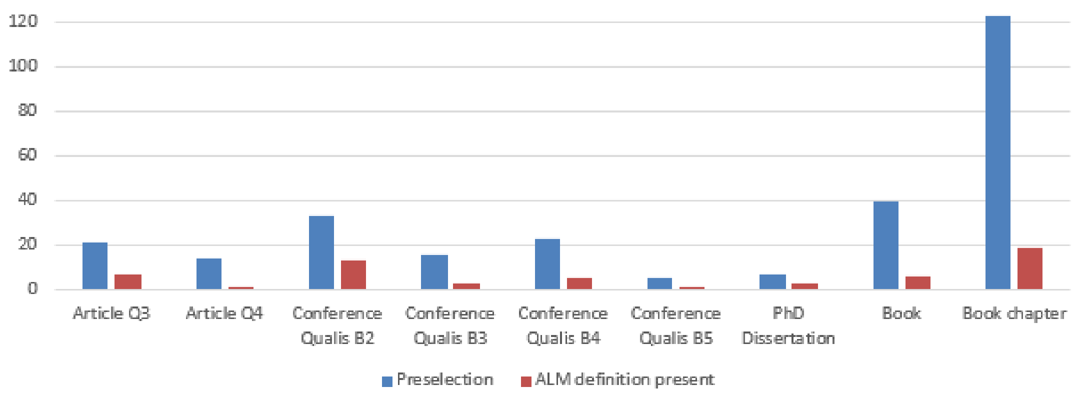


FIGURE 2.17: ALM definition present in preselected extended academic sources

As summary, the PRISMA process provides researchers with a systematic framework for performing a literature review. This methodology consists of four essential steps: identification, screening, eligibility, and inclusion. The data collecting and preprocessing methods employed are depicted in Figure 2.13. The number of papers that have been excluded is also emphasized at each stage using the exclusion criteria.

2.4.6 Critical Review of existing ALM definitions

The efficacy of a critical review is in its capacity to elucidate issues, inconsistencies, or domains where the prevailing understanding of a subject is unreliable. The purpose of this second evaluation is to carefully examine the high-quality literature that has been found on a wide-ranging subject in order to extract and determine a precise definition for the ALM. The review that entails a critical evaluation does not inherently juxtapose the included works with one other. Instead, it evaluates each work based on a specific criterion and determines whether it is acceptable to some degree. By providing a focus and direction for further improvement in areas such as methodology development, ALM can effectively inform other scholars and enhance knowledge development. This is particularly important as several sources describe ALM as a business-related area driven by tool vendors (Markov and Druzhinina, 2011). The establishment of a new study field is frequently motivated by corporate interests. However, the absence of scientific validation for the methodology might become a significant hindrance over time. The objective is to bridge the gap between business and academic stakeholders and create a conducive atmosphere for their collaboration.

For the identified ALM definition sources, a critical review proceeded to analyze and extract the ALM definitions and synthesize their scope that can serve as input for further methodological research.

The critical review process is a valuable approach for assessing and enhancing diverse forms of information. However, it does possess certain limitations, such as subjectivity and a restricted scope, which need to be addressed during the review. While it is important to attempt to discover all the existing literature pertaining to a topic being reviewed, in this particular instance, this can be achieved through thorough reading and organized analysis of the sources. While there is no obligatory mandate to openly describe the methods of search, synthesis, and analysis (Grant and Booth, 2009), the information extraction process is already predetermined to address this vulnerability. The objective is to extract and categorize the definitions from the sources. In order to establish the classification, a scoping method needs to be created after reviewing the sources. According to that classification, will now be proceed by listing and explaining the definitions.

The significance and judgment for the selection of ALM definition in a critical review are outlined below, along with explanations. The objective of this research is to investigate how ALM is being defined, if at all, and to make an effort to define its breadth and understanding.

The results for the Systematic Literature Review and the Critical review are to be found in the Chapter 4.

2.5 ALM logic planning and scheduling

In this section I first clarify the understanding of flexibility and uncertainty, then clarify the types of flexibility. After that defining the content of planning and schedule expressions. As for ALM it is not proper to use the project planning expression due to the scope difference, I will refer to it hereafter as logic planning which is a broader expression.

Uncertainty on one hand, refers to a lack of knowledge, information, or predictability about a future event or outcome. It represents a state of not knowing the exact outcome or being unable to determine the probabilities associated with different outcomes. Uncertainty can arise due to various factors, such as incomplete information, complexity, randomness, or the presence of multiple possible outcomes. It can be part of each plan, in the project management area several scholars were already studying how to handle it (Pich et al., 2002).

Flexibility, on the other hand, refers to the ability to adapt, change, or adjust to different circumstances or requirements. It represents the capacity to modify one's approach, actions, or plans in response to new information, changing conditions, or unexpected events. This means, from the beginning, a change is expected to be handled, and there is a preparation accordingly. E.g., there is an acceptance range defined, for that Pich et al. (2002) refers that it is the manager's job to anticipate it by creating flexible contracts.

In summary, *uncertainty* represents a lack of knowledge or predictability about future outcomes, while *flexibility* refers to the ability to adapt and respond to change.

Managing uncertainty with flexibility is known already from the stochastic production areas, for example, Gerwin (1987) and Morales et al. (2014) describe a case for solar and wind power production facilities. Morales highlighting also that in case flexibility is already included in the planning, uncertainty handling is already prepared. Similarly, for the ALM environment thus I adapt the view to prepare with flexibility in the affected areas to cover the uncertainty also. There are different types of flexibility, timewise, modality, line of activities, cost, and resources. See details in the 2.5.1 subsection for matrix-based scheduling.

Project planning is generally understood as logic planning, i.e., the predetermination of actions, and all the other resources that are necessary to achieve the objectives, applying scheduling to these actions and assigning resources, by which also costs can be determined. According to the PMI (Project Management Institute), the planning process is the defining and refining of the objectives and the selection of the best alternatives to achieve the targeted objectives (PMI, 2021). Laufer and Tucker (1987) has defined project planning as the method of planning, monitoring, directing, communicating, scheduling, and cooperating between the stakeholders, whereas project planning is the formulation of goals and objectives that explain the work that has to be done. The scheduling identifies the timeline assigning the resources that are required (Zwikael, 2009).

In summary, hereafter the *logic plan* defines the structure, i.e., the tasks and their connections with precedence. *Scheduling* represents the timing and resource usage of the defined structure.

After setting the base understanding, the next step is to examine the TPM, APM, and ALM approaches for the logic plan differences. In the case of the TPM, for traditional planning logic planning contains the definition of the activities and their connections (Pellerin and Perrier, 2019). After this, the scheduling can be directly executed.

In the case of APM, after the logic planning, it is necessary to set priorities for the activities/tasks so that the execution can be determined. The tasks with the highest priority (1) are always executed, less than 1 means that activities become *optional*. Enabling the removal of the tasks means implicitly also that the connection between the activities can dissolve. In that extreme case, all the tasks' priority is 1, which means all the tasks are to be executed, then the setup will be the same as in the case of traditional planning.

In the case of ALM, in addition to the APM setup, *additional* activities are expected to appear. This means that preparation for *flexible* handling of such activities is needed. The *additional* activities must be then expected in a predefined way. The definition of the handling must be contracted in advance in the offering phase already. This can result in a predefined amount of *additional* activities that can be incorporated without changing the boundaries. Extending the boundaries is possible also, however, for the handling of the surplus *additional* activities also must be agreed.

After defining the logic structure above, the next step is the examination of the scheduling problem. Let us review the different approaches. In the case of the traditional (TPM) approach, the scheduling can be planned based on the earliest or latest start of the activities. In this way of scheduling, there is no flexibility as a defined point of start is expected. It is possible to have flexibility in case it is expected not a point of time but an interval for starting.

In the case of the APM approach, there can be flexibility coming from the structural planning, i.e., omitting the lower probability of the *optional* activities, which change affects the structure also. Also in the scheduling, there is flexibility, as reorganizing the tasks' dependencies is possible in this scheduling. So overall, there is flexibility in the logic planning and in the scheduling also for the agile approach.

In the case of the ALM approach, above the APM approach, there can come additional flexibility coming from the *additional* activities. This means that unplanned activity handling must be evaluated. The contracts already defined a range for acceptance for the activities when they appear. Over the range, the surplus activities handling can be either accepted and extend the boundaries or declined to be executed. So the additional flexibility factor is coming from the handling of the *additional* activities.

In summary, uncertainty, and flexibility were defined. The model handles both of the approaches regardless of the type, thus the flexible planning application will cover all the cases.

A project scheduling problem is identified as determining the time required to implement the activities of a project plan to achieve the goals. In the primary research, only the execution time factor was considered thus methods like Critical Path Method (CPM) and Program Evaluation and Review Technique (PERT) are proposing the consideration of the precedence. One of the major limitations is here though the lack of resource constraints, which was then resolved later in the Resource Constrained Project Scheduling Problem (RCPSP) (Habibi et al., 2018). Since the 1950s, the resource-constrained project scheduling problem (RCPSP) has been extensively studied in the field of project planning. This classical problem involves scheduling a set of activities, taking into account both precedence and resource constraints, to optimize an objective function such as minimizing the overall project duration or overall costs. Over the years, numerous researchers have devised exact and heuristic solutions for this problem (see Moukrim et al. (2015), Kreter et al. (2018), Tritschler et al. (2017), Abdolshah (2014), Demeulemeester and Herroelen (2006), and they have also explored various approaches and extensions. In their work, Hartmann and Briskorn (2021) offer a comprehensive overview and classification of the most significant extensions of the RCPSP already as a second review since 2010. A very comprehensive, state-of-the-art view of the different methods, variants, features, and objectives is also collected by Sánchez et al. (2022).

The resource-constrained multiproject scheduling problem (RCMPSP), which is an essential extension, focuses on managing multiple projects that share the same set of resources while ensuring that resource constraints are not violated. Since its initial introduction, various researchers have examined different variations of the resource-constrained multiproject scheduling problem. However, it is worth noting that only a small number of scheduling algorithms specifically tackle multilevel projects, and these algorithms typically adhere to traditional scheduling methodologies. In such cases, the execution of activities is still in a fixed order, see Pellerin and Perrier (2019). Recent algorithms usually decompose multilevel projects into collaborative or competitive single projects, that are solved in a distributed way using agents (D. Liu et al., 2019). Nevertheless, these approaches also make the assumption of fixed logic plans for projects. For instance, when it comes to software development projects, they are commonly executed within the context of multiproject environments and exhibit flexibility through the adoption of methodologies like agile, hybrid, or extreme project management (Marchenko and Abrahamsson, 2008).

The most frequently used traditional planning methods are network planning methods, Gantt charts, and Line of Balance methods primarily support the operative tasks of project planning (Zsolt Tibor Kosztyán, 2015). Network-planning methods (see Wiest (1981)) supporting traditional project management approaches only and have several deficiencies and difficulties when using project planning methods, e.g., inability to handle reappearing tasks or projects where certain activities must be skipped due to time, cost, or resource constraints. Network-based project planning does not consider several possible outcomes and does not provide an opportunity to prioritize activities and subprojects. This is why alternative and extended methods are necessary to be considered, such as the matrix-based methods, which are better for identifying and handling reappearing tasks and resource constraints for example (Kosztyán and Kiss, 2011). An additional drawback of the network planning methods is that they lack the support of flexible and agile projects. Therefore additional method was discovered, which is appropriate for flexible project representation, see the matrix-based scheduling in the following subsection.

2.5.1 Matrix-based planning and scheduling

Matrix-based planning can eliminate the shortcomings of traditional methods, next to the traditional projects, it is possible to plan agile and hybrid projects also.

From a representation point of view, projects can be represented as graphs. There are two kinds of approach, the so-called activity-on-arrow networks [AoA] in which activities (or tasks) are depicted as arcs (Demeulemeester, 1996), and the activity-on-node networks [AoN] where activities are denoted by the nodes (Ren et al., 2021). The matrix representation of projects usually describes an AoN network (Minogue et al., 2011).

The matrix-based project planning methods are often based on the design or dependency structure matrix (DSM) (Zsolt Tibor Kosztyán, 2015). The domain mapping matrix (DMM) is an extended version of the DSM, with multiple domains (Danilovic and Browning, 2007). Using the Numerical DSM (NDSM), the level of dependency relationship between two activities can also be plotted (Tang et al., 2010). With the stochastic network planning method (SNPM) developed by Z. Kosztyán, J. Kiss, et al. (2010), probabilities or priorities regarding the completion of the activities can be considered already, enabling various possible network plans to be modeled due to the parallel or sequential completion mode of the tasks. In the case of the project expert matrix (PEM), which was created as a further development of the SNPM, the relationships between the activities can be uncertain or stochastic, as can the completion of the activities in the project scenario. The project domain matrix (PDM) proposed by Zsolt Tibor Kosztyán (2015) is used to cope with multiple domains, and it is an extension of PEM to be able to handle time, cost, and resource demands and constraints. Zsolt Tibor Kosztyán (2015) suggested a project domain matrix (PDM), that can be used for both single and multimodal project plans. PDMs allow mandatory and supplementary tasks with priorities and flexible dependencies between tasks. Kosztyan and I. Szalkai (2020) later extended this matrix-based model to address multiple projects, programs, and even project portfolios. Such a matrix representation, due to the handling of flexibility, seems to be a good base for an ALM problem description, as will be shown in the later chapters.

2.5.2 Flexibility in logic plan and scheduling

From a practical point of view, project managers face the dual challenge of maintaining project focus while also addressing their organization's imperative to introduce changes and uncertainties in the business landscape. In order to optimize the efficiency of project organizations, flexibility is typically discouraged during the later stages of projects in traditional project management. On the opposite, agile and hybrid project management is welcoming the changes, and counted as part of the normal way of working. Consequently, the adoption of structured methodologies for managing flexibility becomes desirable (Olsson, 2006b; Kreiner, 1995).

Projects managed by traditional methods assume that the activities have a fixed order of execution in the project plans (Pellerin and Perrier, 2019). Software development projects and Application Lifecycle Management also have flexible attributes like agile, hybrid, or extreme projects (Marchenko and Abrahamsson, 2008), thus the dependencies of activities are not necessarily fixed (Zsolt Tibor Kosztyán, 2015). The priorities for these tasks are set to select which tasks will be either completed in a short project (a so-called sprint), postponed, or skipped. Agile project management allows such flexible dependencies and priorities of task completion (Zsolt Tibor Kosztyán, 2015), while extreme projects allow new and unplanned tasks for common changes in stakeholder requirements. Hybrid approaches allow traditional trade-off methods besides flexibility with multimode task completions (Zsolt T. Kosztyán, 2020).

Flexible approaches are often used in non-IT development projects also (Hidalgo, 2019; Metzger et al., 2021). For example new product development projects (Ciric, Lalic, et al., 2019; Morales et al., 2014), Research and Development (Huchzermeier and Loch, 2001), construction industry (Arefazar et al., 2022) and maintenance (Z. Kosztyán, Pribojszki-Németh, et al., 2019).

Kosztyán (2022a) proposes the matrix-based modeling of the flexible project structures also next to the traditional. The base for it is a project domain matrix (PDM) which has 3 mandatory domains, namely, logic domain (LD), time domain (TD) and cost domain (CD), and two supplementary domains, namely, quality domain (QD) and resource domain (RD). Their proposed matrix-based flexible project planning (MFPP) tool implements a genetic algorithm-based solver. Since all agents must decide which tasks and dependencies must be included in the project and which completion mode to implement, the result contains neither flexible dependencies nor supplementary tasks or different completion modes.

These results are also represented in a matrix (PSM - Project Structure Matrix) that has 4 mandatory domains, including the LD (Logic Domain), TD (Time Domain), CD (Cost Domain) and SD (Scheduling Domain), where the scheduled start time (SS) is presented, and 2 supplementary domains, namely, the QD (Quality Domain) and RD (Resource Domain). The TD, CD, and QD are still vectors. The PSM matrix already contains a schedule domain of scheduled (in this case earliest) start time. The PSM does not contain flexible dependencies or supplementary tasks because agents decide which tasks and dependencies have to be included or excluded from the project.

Flexibility can be defined from several aspects. In the model, it is handled as follows.

Time-related flexibility exists in the logic plan or structure itself, can result thus from slacks or topological floats (Vanhoucke et al., 2008). In this case, the precedence relations and the implementation modes remain the same, and only the scheduled start and finish times of the tasks change. Hauder et al. (2020) shows how this flexibility can change the logistical (storing or conveying) task duration.

Flexibility in scope occurs as the tasks are defined with a probability in an interval between 0 and 1. In case the activity is not selected to proceed, it will be removed from the plan, consequently causing a change in the schedule also. Thus the omitted activity from the plan influences the schedule. In this case the flexibility is coming from the structure definition, even though the effect is realized in the schedule.

Modal flexibility in which a task can be performed in multiple modes. So the same result can be achieved by carrying out the same tasks with different technology and the related, maybe different, time demands. Extensions to the resource-constrained project scheduling problems (RCPSP) with alternative activity chains (RCPSP-AC) are defined (Tao and Dong, 2017). In RCPSP-AC, there are interchangeable process patterns/processes/activity modules/methods which are called activity chains here. An activity chain includes one or more activities that are related by precedence relations. Each activity chain can be an alternative for other others and only one of them can be selected for execution. The restriction though from the ALM perspective is so that these alternative ways must be already in the planning phase defined. In case of an unforeseen activity rising in the ALM environment during execution, that cannot be predicted in the planning phase yet.

Dependency flexibility is an additional type. Some logical dependencies can be omitted in case the technology necessary for the activity does not require a strict sequence. Omitting a dependency lifts the restriction of sequential execution and allows the associated tasks to be performed in parallel or an arbitrary, relative order.

Flexibility in cost and resource planning in the current model not yet considered. Use-case can be for example a change in the used raw materials or its availability in the market resulting in cost or processing effect. However, the extension is possible for future research.

2.5.3 ALM scheduling problem overview

For the ALM scheduling problem definition in the academic literature, there are not yet available recommendations and studies from a methodology or scheduling point of view. Lacking the proper understanding of the ALM concept and the scattered attribute of the development phases (Sonnemann et al., 2015a; Kääriäinen, 2011), several times it is forced into the framework of project management or service management. Jamous et al. (2016) claims also severe improvements are expected in the handling in the area still to improve for ALM-specific environments.

The background work in the previous chapters was necessary to clarify the flexibility of conditions and applicability of boundary extension from the project management approach. ALM scheduling has similarities with project management in the sense of development, in case the time window applied for ALM is narrowed down. However, the significant difference in ALM compared to project management approaches is that additional, unplanned tasks must be handled, which were and could not be planned during the contracting phase yet. For the contracting period in classic project management generally where the scope, price, and duration were clearly defined, and there was little room, if any, for deviations. In the case of an ALM, these are also not available, it is rather possible to define intervals, and set boundary conditions in handling of activities. ALM is more familiar with service management handling, for example, SLA (Service Level Agreement) about the Quality of Service towards the content of development. Such boundary definitions are to be handled in the contract already upfront with flexibility (Barata and Camarinha-Matos, 2002), e.g., to define the applicable maximal resource usage, applicable payment for overtime handling, flexible resource involvement options for time and cost limitations, review milestones (Ng and Navaretnam, 2019). Other academics already proposing also periodical reviews for contract management, to keep the competitive advantage in each lifecycle (Algarni, 2021). Such contracts can be called also LCC (Life Cycle Contracts), DBFM contracts (design, build, finance, maintain) or DBFO contracts (design, build, finance, operate) where the whole lifecycle of the product is covered in the model (Ilin et al., 2022).

During the research of the literature, ALM characteristics were following the organic, flexible project structure modeling rather than the mechanic, where mechanistic reflects the traditional, i.e., waterfall approach, and the organic reflects a more adaptive approach, higher awareness of dynamic project environment, and changing requirements characterized by flexibility (Sohi et al., 2019).

For Flexible project schedules, there are novel methods available in the recently published matrix-based solution by Zsolt T. Kosztyán (2020). This means the application base from flexible projects is established and validation of the model in the ALM environment is reasonable also.

2.6 Success evaluation and Risk handling

In this section, after a short introduction for success understanding the main focus is on the risk and risk handling in classical project scope, I highlight the challenge for risk factor identification for ALM, then provide a literature review summary understanding the main relevant risk factors for ALM. Then I provide insight into the results from the simulation-related risk factor identification and significance.

2.6.1 Success evaluation

In general the assessment of success in software project development involves several aspects, such as project goals, stakeholder satisfaction, schedule and budget control, product excellence, process effectiveness, risk handling, value provided, adherence to regulations, and team proficiency. Success is generally determined by the degree to which the project fulfills pre-established criteria and deliverables, while also keeping to specified time-frames and budgets. This classical approach is based on the iron triangle (time, cost, quality) perspective fulfillment and very popular in the traditional project management literature, however, several academics are debating the timeliness of this limited view on success, thus proposing to exchange the quality for example with Scope, Performance, or Requirements to get a more clear view on the success aspects (Pollack et al., 2018).

Regular feedback and involvement are crucial for assessing stakeholder satisfaction. Quality measures, such as the rates of defects and performance indicators, together with adherence to best practices, are of utmost importance. Efficient risk reduction and resolution of problems further indicate the achievement of project goals. The project's performance is highlighted by the delivery of strategic business value, which is frequently assessed using metrics such as Return on Invest and business impact. In addition, regulatory compliance and the enhancement of team competencies have substantial importance. Evaluation approaches, such as Key Performance Indicators (KPIs), balanced scorecards, post-implementation evaluations, and earned value management, offer complete frameworks for systematically and methodically measuring the effectiveness of a project (Garousi et al., 2019; Agarwal and Rathod, 2006),

In contrast to evaluation of the project feasibility, predicting the success of a project is a more challenging task. Although the feasibility of project plans is satisfied if they fulfill the requirements, for the success of a project, it is also essential to satisfy the stakeholders and to meet their predefined expectations with the project outcome. Therefore, when specifying a model to predict the project success, it is important to score not only the realized tasks but also the whole project outcome.

The perception of success (or failure) could be divergent for different stakeholders, and this perception could also change over time. Hughes et al., 2016 concluded the lack of a uniform set of acceptance and success criteria that is applicable for every project. In terms of the success of the project management, the technical side, namely, cost, duration, and quality/scope, is often complemented by soft factors and skills of the people-related side, namely, political, social and cultural issues, how the delivered product is welcomed and used, and how well the results dovetail with the strategy of the sponsor/project owner/customer.

Hughes et al., 2016 also state that despite several studies ascertaining that the reason of failure emanates from poor project management, they omit to indicate that this originates from incorrect use of the methodology or that the method itself is unsuitable for that particular project. In this work, a vendor perspective used and only deal with the project management success, the (hard) technological aspects of project performance regarding the duration, cost and quality/scope requirements, and its link to the applied project management methods. This approach is parallel to the concept of efficiency by DeToro and McCabe, 1997, where the authors distinguished the efficiency (the project meets all internal requirements for cost, margins, asset utilization, and other efficiency measures) from the effectiveness (it satisfies or exceeds all customer expectations).

Complex project success understanding existing in the literature like (Görög, 2002), who are claiming several aspects to consider like efficiency, customer impact, stakeholder satisfaction, business success among others. Such complex approach might have future adaptation for the ALM scope, however, currently in the ALM area there is no such literature available yet.

Since the satisfaction with the expected outcomes can only be estimated, the extended sense of the project feasibility is used instead of project success. However, the used model is closer to the quantitative definition of project success than that of traditional feasibility because the model considers the customer's stated requirements through the project score.

This approach regards the project plan as infeasible or failed if *Process failure* (i.e., the system delivery has failed to meet its defined criteria in terms of time, budgetary constraints, and schedule) or *Correspondence failure* (i.e., the system does not meet all of its goals and quality criteria and the implemented system does not correspond to the requirements) occurs.

One of the most interesting results of the Chaos Report (SGI, 2019) was that the IT projects managed by agile project management approaches were 3 times more successful than traditional or waterfall-type projects. This considerable result is confirmed by other surveys and scholars (see, e.g., Wysocki, 2011a; Dan, 2016). However, hybrid approaches, such as the combination of traditional and agile project management, are not explored, while several case studies see, e.g., Dove et al., 2018; George et al., 2018, surveys see, e.g., Duka, 2012; West et al., 2011 and studies see, e.g., Leybourne, 2009; Wagenaar et al., 2018 indicate that usually a mixture of traditional and agile approaches are followed. Although Pich et al., 2002 and Sommer et al., 2009 have stated that there is no superior project management approach for all kinds of projects.

So for the project success in this thesis rather the classical iron triangle approach is favoured, the fulfillment of the time, cost, resource goals are evaluated and with score points are rewarded, the overshoots are penalized. As success evaluation was not the primary target of this comparative analysis of methodologies, the present concept is only limited, however, future extension might be desirable for further improvement.

2.6.2 Project risks and risk management

Risk is characterized as the absence of assurance regarding the outcome, which can either be a positive change or a negative threat. Effective risk management involves the process of recognizing and regulating any hazards that could hinder an organization's ability to meet its business goals (Government Commerce, 2007).

Identifying risk is a complex task in project management. Many project managers often spend insufficient time to identifying risks and instead focus more on risk mitigation. However, they overlook the reality that risks that have not been recognized cannot be effectively addressed. Nevertheless, it is important to understand that the process of identifying risks is fundamental to establishing of an appropriate risk management strategy. Once the risk has been identified, a systematic planning approach may be implemented to effectively deal with it. The process of identifying and managing risks may be a challenging endeavor that necessitates the involvement and assistance of all those involved in the project. Furthermore, the project manager does not possess exclusive information and thus relies heavily on input from other stakeholders, particularly in the identification of project risks (C. George, 2020).

Managing risks are standard task in project management already, the intent behind Risk Management is to identify, evaluate, analyze, assess, and mitigate potential product issues defined in ISO/IEC 31000 (Barafort et al., 2019) also. Risk Management is a total product life cycle process according to PMBOK also (PMI, 2021).

2.6.3 ALM risk understanding overview

Risk is normally perceived as something to be avoided because of its association with threats, and as previously introduced, the ALM environment is more extended compared to the project scope, thus it provides additional space for potential risk factors to appear. Unfortunately, the risk factors for ALM are scarcely researched yet, the literature mainly contains narrowed-down ALM scopes. In the following, those ALM environment-related risks are presented, which are identified from the structural and scheduling point of view from the available academic literature. This means that the general, e.g., ALM organizational point of view is disrespected here, even though there are significant risk factors also identified for ALM organization adaptation (Akgun et al., 2020; Tüzün et al., 2019), and later on related to operation (Cheng, 2010).

Risk management approaches are also different for Agile, which is often used in the ALM environment, as the intention of Agile ideology with the iterative loops is to "fail early" and react to the issues. Buganová and Šimíčková (2019) creates an analysis to compare traditional and agile risk management and highlights the advantages and disadvantages on both sides. She points out that organizations use projects to manage changes for developing and deploying new products. In today's competitive environment, only those who can manage the risks and realize the project more efficiently will succeed.

Due to the above-discussed differences in ALM and PM scope, the risk scopes require additional analysis. Project- and SW-wide risks also need an extension in theory for the ALM scope. Academic research for this field is very limited, a risk collection and assessment tool is proposed by Choetkiertikul and Sunetnanta (2012), mostly focusing on distributed SW development-related risks. However, mostly the general Life Cycle Management area risk management (Sonnemann et al., 2015b; Hummer et al., 2019; Niemann and Pisla, 2018; Castaneda et al., 2020) or the Software Development Life Cycle is researched (Sahu et al., 2014; Roy, 1962). So in the following, as a restriction to current understanding, the relevant risks will be treated such as project risks, which should be proper and acceptable for the ALM model. The limitation can be resolved with a further study of the ALM scope in the future.

Table 2.9 collects the factors from the literature related to the project management and Application Lifecycle Management risk factors, and showed which academics were investigating on the topics. Below I am providing a brief insight also how they are related to general project approaches, SW projects, and ALM. Since the focus of this dissertation is the methodological approach, the main emphasis thus is on the ALM-specific non-planned activities elaboration and its effects.

Scope Creep. Komal et al. (2020) indicates that scope creep is present mainly in SW projects, and investigates with a thoroughful SLR their reasons. According to him, software engineering and software project management experts in the literature have asserted that scope creep is a prevalent factor contributing to the failure of software projects. Furthermore, critics assert that it has the potential to manifest in nearly every software project, resulting in a compromise in quality, delayed schedules, escalated costs, and diminished client satisfaction. Madhuri et al. (2018) investigates also scope creep for project scope creep in SW companies and takes one more step to visualize and propose its management using a mathematical modeling perspective in leading SW companies, respectively Ajmal et al. (2022) in the construction industry. For ALM, the Scope creep due to the execution of the non-planned task is undoubtedly present, as Rossberg (2019) highlights in his book as an ALM-specific factor that stakeholders must pay attention to and manage. Aiello and Sachs (2016) even proposes ALM agile methodologies and the utilization of DevOps for preventing risks related to scope creep.

Change in requirements. Project management based on the requirements management in the upstream, and in the traditional approach later changes are not welcome. In project management practical requirement handling is a key factor for projects, what Kossmann (2016) also describes and explains in his book. Venkatesh and Balani (2016) highlights that Requirement management is a key to successful SW projects also. Due to the fact that the non-planned activities with a high chance have also requirement changes, in ALM, this risk is also present, even with a much higher occurrence rate than in project management approaches referred by Chanda et al. (2013) and Rossberg (2019).

Risk Factor	Presence in Project Management	Presence in ALM	Primary in ALM
Scope Creep	Komal et al. (2020) and Madhuri et al. (2018) Ajmal et al. (2022)	Aiello and Sachs (2016) and Rossberg (2019)	No
Change in Requirements	Kossmann (2016) and Venkatesh and Balani (2016)	Chanda et al. (2013)	No
Budget Overruns	Jackson (2002) and Albtoush and Doh (2019)	Ebert (2013) and Banjanin and Strahonja (2018)	No
Schedule Delays	Majerowicz and Shinn (2016) and Park (2021)	Tudenhöfner (2011) and Aiello and Sachs (2016)	No
Resource Constraints	Mishra (2020) and Issa and Tu (2020)	Rossberg (2019) and Rossman (2010)	No
Feasibility of problem	Issa and Tu (2020) and Rahman et al. (2021) Beek et al. (2024)	Aiello and Sachs (2016)	No
Quality Issues	Komal et al. (2020), Shafqat et al. (2022), and Wawak et al. (2020)	Otibine et al. (2017) and Akgun et al. (2020)	No
Lack of Traceability	No	Corallo et al. (2020) and Akgun et al. (2020)	Yes
Version Control Issues	No	Kääriäinen and Välimäki (2008) and Pirklbauer et al. (2009)	Yes

TABLE 2.9: Risk factors appearing in Project Management and Application Lifecycle Management environments

Budget Overruns. In classical project management, the increased costs are mostly influenced by improper planning, but also by the non-panned and non-compensated activities, which makes it similar to the situation in an ALM environment. Jackson (2002) and Albtoush and Doh (2019) in the construction industry checking the cost overran risk factors and evaluating their handling. The main finding is that improper change management and risk handling can lead to overshoots. Ebert (2013) highlighting the advantage of an ALM system in general for improving the efficiency of a product or SW development, thus managing the budget also. Banjanin and Strahonja (2018) above the risk factors investigation in the ALM area also proposes a framework to reduce risks for budget overruns on the portfolio level already.

Schedule Delays. Majerowicz and Shinn (2016) investigates the correlation between schedule delays and expense overruns in complex projects. Many project practitioners commonly agree that cost overruns are directly correlated with schedule delays, however, this is not a hard fact. Similarly, Park (2021) was proceeding with a study that examines in classical project management the occurrence rate, extent, and attributes of schedule delays that take place during the building of 113 sizable construction projects, finding factors, like non-planned activities can have a significant effect on cost overran and schedule delays. For ALM advantage Tudenhöfner (2011) reveals that while the traditional projects the schedule is delayed if an unpredictable issue occurs that must be corrected, for ALM the framework provides a higher level of flexibility in planning and with its integrated performance management. In opposition, Deuter, Otte, et al. (2019) puts parallel the ALM and PLM integration for scheduling activities, making the integration of two approaches proposed.

Resource Constraints. Most of the projects have some kind of limitations in their resource availability, and handling resources in an efficient manner is a cost-sensitive target for the projects in classical project management. Mishra (2020) is investigating resource usage in a road construction environment utilizing the Theory of Constraint (TOC) and Critical Chain concepts to improve overall performance for effective scheduling. He is proposing for non-planned activities the handling with a buffer strategy. The study of Resource-Constrained Project and Multi-Project Scheduling Problems (RCPSPs and RCMPSPs) has been crucial in the past thirty years. Both problems involve the arrangement of activities, taking into account their order and limitations on available resources (Issa and Tu, 2020). Also present dissertation examines an extended RCPSP matrix representation method for ALM problem solution. Aiello and Sachs (2016) refers in his book for ALM that it helps to deal with the shifting priorities by clarifying the resources required for non-planned tasks and their effect in the schedule (Rossman, 2010; Rossberg, 2019).

Feasibility of Scheduling The Resource Constrained Project Scheduling Problem (RCPSP) for project management schedule feasibility analysis is a well-known area. Several academics were investigating the extension of the base problem for real-life-like approaches (Issa and Tu, 2020) for scheduling disturbances or delays(Rahman et al., 2021), flexible structure (Van der Beek et al., 2022; Beek et al., 2024) or unplanned tasks appearance (Kosztyan and I. Szalkai, 2020). For the ALM area, the non-planned tasks are expected within the scope of the problem area to be able to be dynamically handled. This is happening with flexibility in the schedule adaptation in an agile way usually (Aiello and Sachs, 2016).

Quality Issues. (Komal et al., 2020) was already highlighting that scope creep, which means the constantly increasing content, can be e.g., the unplanned tasks appearing in the project, can be the cause for loss of original project targets such as quality also next to the schedule and cost. Shafqat et al. (2022) is making an approach how to plan the unplanned activities to manage quality levels, in his case for design iterations to keep quality high. He examines how companies in New Product Development are handling 'proactive risk management' and 'reactive fast learning'. Wawak et al. (2020) was proceeding with an SLR for construction industry quality-related main factors to identify, and found that most mentioned product quality factors are "compliance with scope". This indicates that the non-planned activities have a significant effect on the quality.

For ALM, Otibine et al. (2017) was investigating the question of the quality correlation, however, he found that ALM solutions prioritize the integration of software development phases, but do not adequately address the topic of quality. The concept of quality has been kept vague. Akgun et al. (2020) stating that ALM main purpose is to improve software quality. However, direct discussion about the appearance of additional non-planned tasks evaluation is not present in the detailed discussions neither here nor in other ALM-related articles.

Lack of Traceability. While in classical project management, it is not necessarily part of the scope, for ALM, traceability is a key aspect (Corallo et al., 2020), proceeding with a SLR to discover the connection between traceability and lifecycle in six industries (Software, Manufacturing, Automotive, Automation, Aircraft, and Aerospace). Akgun et al. (2020) also highlights the main advantage for ALM the traceability availability ab ovo functions from vendors.

Version Control Issues. Version control is tightly integrated in ALM, however, it is not necessarily part of the classical projects (Kääriäinen and Välimäki, 2008). Throughout the lifespan of an application, several versions emerge and require systematic control for overseeing releases, preserving predetermined states and benchmarks across different components, and returning to these predetermined states as needed. The concept of version control is widely recognized, with ongoing research expanding the scope of version control beyond source code artifacts (Pirklbauer et al., 2009). Due to an additional task appearing in the version control system also involvement is necessary, which enables the proper artifact and process tracking. However, improper handling of version control can lead to conflicts, loss of data, or unintended overwrites.

So as summary these were the risk classification related literature research evaluation for the project and ALM environment. It was evident there are several risk factors are also considered in the ALM also in an extended sense and there are new considerations appearing as well.

2.6.4 Matrix-based risk management

Failing to understand and manage (software) project risk can lead to a variety of problems, including cost and schedule overruns, unmet customer requirements, and products that are not used or do not deliver business value. In accordance with the ISO 31000:2018 (ISO, 2018), it is used the term risk regarding the effects of uncertainty on the objectives that result in a deviation from the expected.

When managers deal with risk, they seek to influence their environment to reduce negative outcomes (Wallace et al., 2004). Advocates of software project risk management suggest that project managers should identify and control these factors to reduce the chance of project failure.

Studies in the last two decades have described many risk management methods. Elsawah et al. (2016) adopted a risk matrix combining probability and the influence of expert judgment. Chatterjee et al. (2018) integrated fuzzy logic and the analytic hierarchy process (AHP) in the risk evaluation of projects and project portfolios. Concentrating on agile projects, Odzaly et al. (2018) developed an agent-based risk tool that identifies, assesses, and monitors risk. In open source risk management software, Ponsard et al. (2019) incorporates Monte Carlo simulation and AHP to evaluate and prioritize risk mitigation measures. Since the agile approach divides the project scope into small pieces, risk identification, and assessment are more frequent than the assessment in the initial and planning phases of the traditional project management approach. Fu et al. (2012) built a matrix-based risk evaluation method that subsequently models the possible interdependencies between risk factors. By the simulation, the chosen method combines the Monte Carlo simulation, risk factor interdependencies, and risk evaluation in contracting and planning as well as during the tracking phase.

The matrix-based simulation method combines the Monte Carlo simulation, risk factor interdependencies, and risk evaluation in contracting and planning as well as during the tracking phase.

Risk evaluation and analysis methods (X. Liu et al., 2013; Oh et al., 2012; Hu et al., 2013) focus on the effects of changes in project parameters, such as changes in demands of resources, time and cost; however, none of these methods addresses the modeling of the change in customer requirements regardless of its high impact on project success, particularly in the case of IT and R&D projects and portfolios see, e.g., Dvir and Lechler, 2004.

Even if they treat the changes in the customer's requirements through a software development project, these methods focus only on the risk factors; therefore, these methods do not model the connections among project objectives, stakeholders and risk factors.

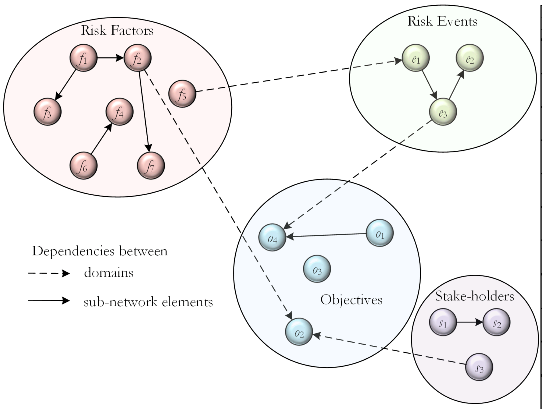
To the author's knowledge, this is the first exposition that applies the so-called meta-network analysis (MNA) technique Zhu and Mostafavi (2015) and Wang et al. (2018) for managing software development projects. MNA can dynamically model the dependencies and interdependencies between and among the following: risk factors, such as delays and cost overruns; the objectives, such as goals and requirements of stakeholders; and the stakeholders themselves, such as project managers (i.e., different kinds of agents), developers, and customers. The original version of the MNA specifies deterministic connections among risk factors and risk effects and objectives. Nevertheless, in an agile and hybrid project environment, almost everything is flexible, such as the dependency between tasks, task occurrences, and project objectives. For example, depending on the implemented project management approach, lower-priority tasks can be excluded from a project; time delays of excluded tasks have no impact on project duration, and thus, the corresponding nodes of the meta-network need to be removed or disabled. The proposed matrix-based version of the meta-network analysis consequently treats stochastic connections between elements.

To model the changes in customer requirements, the proposed framework simulates changes in the score (i.e., priority) of task completions, in addition to the uncertainties of the project parameters. This solution proposes agents (software algorithms to model project management approaches) to manage the aforementioned different kinds of changes and to try to maintain deadlines and budgets simultaneously. The simulation results show which approach should be used to manage various projects and different kinds of risk effects.

Similar to the emergence of matrix-based project planning techniques, matrix-based risk management techniques have also been developed. Fu et al., 2012 and Fang and Marle, 2012 proposed a matrix-based model to analyze the impact of risk propagation and evaluate the resulting risks. These methods subsequently handle the interdependencies between risks; however, they cannot treat the dependencies between risk factors and risk effects or the dependencies between risk effects and objectives. To date, a matrix-based representation of meta-network analysis has not been used. Just as network-based project management techniques are generalized by matrix-based techniques and introduce flexible relationships, Section 3.5.2 reveals that matrix-based risk management techniques can also generalize the network-based risk management techniques, such as meta-network-based techniques.

In the case of the matrix representation of MNA, a multiple-domain matrix (MDM) technique (Eppinger and Browning, 2012) should therefore be used. In this case, MDM is an adjacency matrix of the MNA, where domains on the diagonal represent the dependencies within each subnetwork (i.e., domains) of the meta-network. Such subnetworks are the set of risk factors ($f_1, .., f_n$, e.g., the changes in time, cost and resource factors of a given task); the set of risk effects ($e_1, .., e_m$), such as an overrun of the project duration and budgets; the set of objectives ($o_1, .., o_k$), such as the minimal project duration, minimal cost demands, and maximal quality; and the set of stakeholder requirements ($s_1, .., s_l$), such as the maximal number of WIPs and the maximal project scores. The off-diagonal domains can represent the interdependencies between subnetworks (see Figure 2.18).

FIGURE 2.18: Meta-network analysis and its matrix representation ('X' represents the arcs (i.e., connections) between nodes (i.e., variables)



(A) Meta-network for risk evaluation

(B) Multidomain Mapping matrix for Meta-network analysis

MDM	f_1	f_2	f_3	f_4	f_5	f_6	f_7	e_1	e_2	e_3	o_1	o_2	o_3	o_4	s_1	s_2	s_3
f_1		X	X														
f_2							X					X					
f_3																	
f_4																	
f_5								X									
f_6				X													
f_7																	
e_1										X							
e_2																	
e_3									X					X			
o_1														X			
o_2																	
o_3																	
o_4																	
s_1																X	
s_2																	
s_3												X					

The matrix-based representation can specify not only a binary dependency between nodes but also the off-diagonal cells that can take values on the interval [0,1] and can model the risk propagation between risk factors or, e.g., between a risk factor and a risk event.

In the proposed matrix representation of MNA, the diagonal values can also be specified as a priori probabilities, and the so-called conditional risk values can be simulated or calculated by Bayesian logic. With a two-step Monte Carlo analysis, first, the risk factors are selected according to their a priori probabilities, and only the selected risk factors and their dependencies are considered at the next phase.

Summarizing the literature review, it was presented the project and project management related strong connection with the highlights of the differences and novelties in the ALM environment. It was shown, where are the connection points with the classical project understanding and also it was shown why is not different, like in the content and timing understanding. The ALM specifics were also demonstrated strongly related based on the Software Development attributes, however, on a higher abstraction level representing the entity, highlighting the marginal research material availability for this area which is getting more and more importance with the spread of the even wider software application usage in modern industry. After that the project scheduling methodologies were presented, to get an understanding of their specifics for the analysis upcoming in the next steps.

As highlighted in the theoretical part, the ALM is a relative new area and still lacking common agreement for concepts also, thus a systematic literature review is presented to support the research by finding the relevant ALM definitions. Then the ALM and its context was introduced, and its evolution within the last decades. Next to academic aspects, also the vendor perspective was shortly introduced as a strongly deterministic factor for ALM.

Then the scheduling related matrix planning model and the ALM scheduling problem overview was presented.

Finally the shortly the ALM success and risk understanding was presented with the extended understanding for the ALM environment.

The presented detailed literature review must provide a proper understanding for the ALM scope and novelty of the area, to be able to proceed with the ALM scheduling problem investigation in the upcoming chapter.

Chapter 3

Research Framework and Methods

In this chapter the research details are presented, such as the research assumptions which are articulated based on the presented literature review related to the research questions. After that research framework i.e. simulation environment related information is described that are used for the scheduling evaluation for the project management approaches (traditional, agile, hybrid). The input information for the simulation, like the Data Sources, the different stages of the simulation. After that the used metaheuristic optimizations are introduced. Finally in the chapter the validation tool, the case-study details are shared with the reader.

3.1 Research assumptions

By re-examining the research questions established in Section 1.4 and conducting a thorough evaluation of the findings and connections presented in the existing literature, it becomes feasible to develop the corresponding research assumptions. The research assumptions are outlined below:

> **RA1:** A planning model can be identified based on the unified ALM definitions from the scientific literature.
>
> **RA2:** The TPM, APM, HPM project management approaches using the matrix-based planning method can be extended to solve the scheduling problem, and result in feasible solutions with different results in the ALM environment. A simulation framework can be constructed to handle flexible dependencies and non-planned tasks.
>
> **RA3:** There are existing project-related risk factors that can be extended for ALM scheduling problems, however, due to the differences between project and ALM scope, ALM-specific risks appear also, which can have an effect on resources, cost, and timing, and can influence the feasibility and scheduling performance.

RA1 is necessary to have a connection base for the ALM definition and the applicability of the project management approaches. Based on the literature review and study proceeded it can be assumed that it is possible to sustain this. **RA2** is focusing on the feasibility as main point in the ALM environment. Finding an optimal solution would be a NP-hard task. In the frame of this dissertation, a near-optimal solution suffices to prove the point. Extending risk factors in **RA3** for ALM is necessary to validate due to the differences in the Project Management and Application Lifecycle Management context. The scope of the already existing risks and newly appearing risk factors are also playing an important role.

In the following chapter, the used dataset and methods will be demonstrated.

According to the research questions, the main goal was to compare project management approaches in order to study how they fit in the ALM environment (**RQ1**), and how they are performing in this area (**RQ2**). Furthermore, (**RQ3**) is to determine which risk factors are present in the ALM environment.

Factor	PM	ALM	Constraints
Time	Defined Start and End	Not well-defined	Fixed time window to introduce
Cost	Defined and Limited for the project time	Continuous billing	Budget to define for a specific time window
Resources	Limited and defined if well-planned usage	Limited and defined with sporadic usage	Limited and defined for a specific time period
Unplanned tasks	Not expected	Expected	Flexible project structure

TABLE 3.1: Compatibility overview of main factors for PM methods application for ALM in flexible structures

As Application Lifecycle Management from the previous chapters is concluded with their specific structure, which is to be evaluated if it can be represented in a flexible, matrix structure. The next step shows how to formulate the model that can operationalize the problem so that it can be used for further scheduling analysis with the project management approaches represented by agents.

3.2 Agent-based implementations

During the execution of a project, the project manager typically observes and controls the decision-making process for managing activities. The project manager has the authority to determine the timing and need of task execution. In order to build a simulation, it is also necessary to represent the decision-making process. In this situation, software agents, which are computer programs that act on behalf of a user, were employed to act as the project manager. However, the capabilities of such an agent are significantly restricted in comparison to a human project manager. An agent is only responsible for making decisions related to task execution using defined approaches.

Formally, in TPM, there is a scope that has to be achieved within a given time and a certain budget, but time, cost, and quality can vary according to requirements. The objective function could be the minimal total cost, maximal quality, balanced resource demands, or minimal project duration (see, e.g., Brucker et al., 1999), in addition to the goals in Figure 3.2. TPM can apply different kinds of trade-off methods to balance time/cost/quality/resource demands (Monghasemi et al., 2015). The extension of the trade-off problem, which was implemented as a TPM agent, does not require the trade-offs between resources (Creemers, 2015). This so-called multimode resource-constrained project scheduling problem (MRCPSP) only specifies so-called technologies or completion modes that contain different time, cost, and resource demands for every task. In this paper, this algorithm was used to implement TPMa.

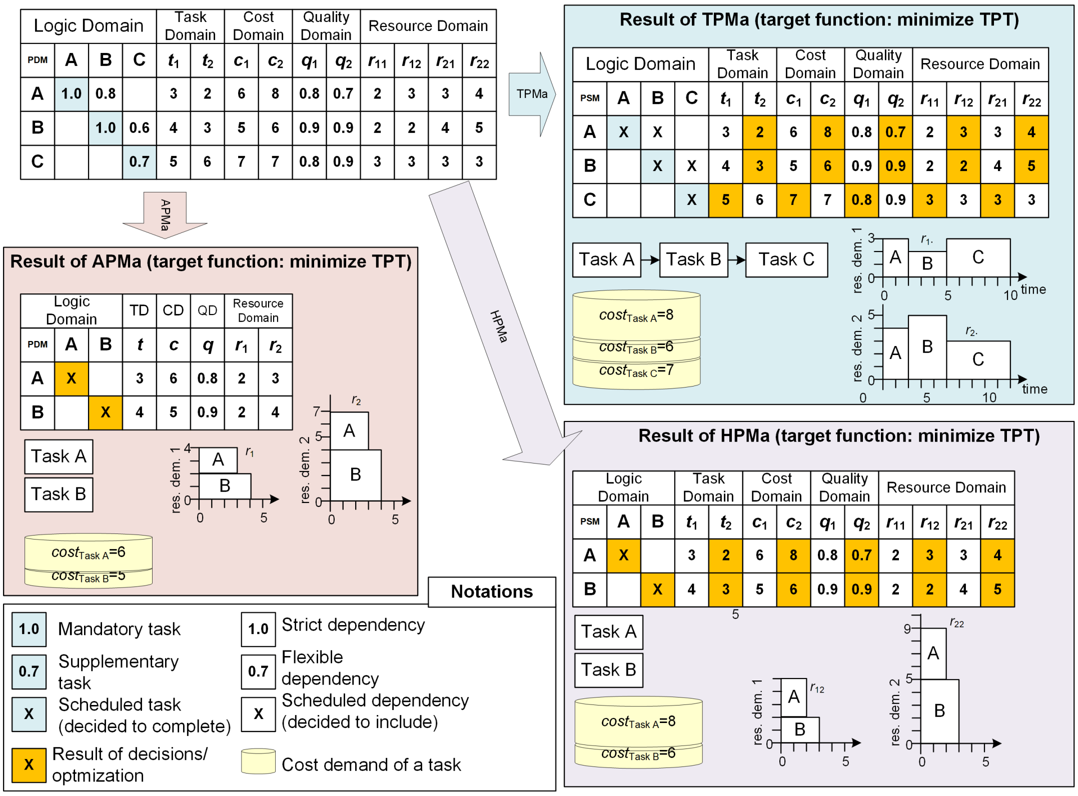


FIGURE 3.1: Comparison of project management approaches and their agent-based implementations when the target function is the minimal total project time. $(t_j, c_j, q_j$ represent time/cost demands/quality parameters, respectively, of completion mode j, r_{ij} is the resource i of completion mode j.)

In the case of MRCPSP, the objective function could be the minimal total cost or minimal project duration. Since trade-off and MRCPSP models have been developed for traditional management approaches, they work in a fixed logic structure. For agile project planning deadlines, resource and cost availability are fixed (see Dalcher, 2009) (see Figure 3.1), and the project structure may be more flexible. The goal could be the realization of as many tasks as possible regarding the importance of realizations and the flexibility of project structures. Nevertheless, minimized total project time, minimal total project cost or balanced resource demands are also relevant target functions for the agile project management approaches (see Figure 3.1).

All the MRCPSP (Creemers (2015)'s algorithm, hereafter TPMa), agile (Zsolt Tibor Kosztyán (2015)'s algorithm, hereafter APMa) and hybrid (Kosztyán and Szalkai (2020)'s algorithm, hereafter HPMa)) scheduling methods are regarded as simplified models of project managers' decisions, and each is realized by a computer program (agent).

In terms of scheduling, traditional time–cost trade-off problems support traditional project management approach (TPMa) and are usually not, or only slightly, considered in agile project management approaches (APMa) (Z. Kosztyán and I. Szalkai, 2018)). In addition, other flexible approaches, like the HPMa, have a flexible structure but can apply traditional trade-off methods and/or multimode task completion (or alternative technology). Table 3.2 compares the selected project management approaches in terms of scheduling. The first feature of the comparison is the project structure, where a fixed project structure means that the structure of the project plan cannot be changed during project completion: a new project plan must be specified to complete the remaining tasks. A flexible structure means that the structure of the project can be reorganized considering the priorities of task completions based on customer preferences. The second comparison feature is the inclusion of new or additional tasks: if it is allowed, new tasks can be included during the completion of the subproject; otherwise, new tasks can be considered only in the next subproject (i.e., sprint). The last comparison feature is how multiple completion modes (or in other words alternative technology) are handled. If multiple completion modes (technology) are not involved, then only a single completion mode can be defined for each task, while if multiple modes are allowed, the project manager (in this case, the agent) can choose the appropriate technology for completing the task. Flexible approaches, such as agile, extreme, and hybrid project management, allow for flexible dependencies between tasks. Because of such flexible technologies, different completion modes, such as serial and parallel completion, can be utilized, and the realization of a given project depends on its constraints. Neither the TPMa nor APMa allows new tasks to be added to a running project: TPMa and APMa postpone these tasks to the next project or sprint. By contrast, the extreme and hybrid approaches can include new tasks if they are within budget. Both agile and extreme approaches consider single completion modes, that is, one possible technology at a time; while traditional and hybrid approaches implement predefined technologies (Kosztyán, 2022b).

TABLE 3.2: Comparison of various traditional and flexible project management approaches

Approaches	Project Structure	New Tasks	Multiple modes
Traditional (TPMa)	Fixed	Not Allowed	Handled
Agile (APMa)	Flexible	Not Allowed	Not Handled
Hybrid (HPMa)	Flexible	Allowed	Handled

In this dissertation, the software agents imitate project managers, the real decision-makers, who have to organize the project within the constraints defined as the scope (see an example in Figure 3.1)

Figure 3.1 shows a comparison of project management agents through an example, where the target function is the minimal TPT. The figure shows that even if constraints are not defined, different results can be obtained with different agents.

These computer programs (agents) are based on scheduling, cost-minimizing and resource allocation algorithms. These agents aim to specify a project scenario from a stochastic project plan that is feasible in the extended sense (there exists a solution within the given boundary conditions). This project scenario can be represented by a project domain matrix (PDM) (see the example in Figure 3.1).

The *traditional project management agent* (TPMa) can use the traditional time/cost trade-off or multimode resource constraint project scheduling methods in order to reduce the time and/or cost demands (see, e.g., Creemers, 2015) of the project (see Figure 3.1) and can use resource allocation and/or resource leveling algorithms for specifying a time and/or resource-constrained resource allocation if it is necessary, but the logic plan of the project is fixed (see the results of TPMa in Figure 3.1) and independent of the task priority. Therefore, the project plan will not be restructured. Unfortunately, in the scenario of applying the trade-off and MRCPSP methods, the time/cost/resource demands cannot be decreased sufficiently without restructuring the project plan (Zsolt Tibor Kosztyán, 2015).

The *agile project management agent* (APMa) can ignore supplementary task completions (see the results of APMa in Figure 3.1) and it can restructure projects if the uncertain task dependency is ignored. In this way, the logic plan can be restructured considering the management requirements (see, e.g., Zsolt Tibor Kosztyán, 2015). Nevertheless, in the restructuring, the lower priority but otherwise important tasks might not be completed, which can reduce customer satisfaction.

However, when running a sprint, unplanned new tasks and new requirements can be involved only until the next sprint. The extreme project management (EPM) approach handles the new tasks and new requirements during the implementation of the project. Extreme project management can confirm the extra costs and the increased project duration due to the extra tasks.

Hereafter, the algorithm for solving the hybrid multimode resource-constrained project scheduling problem (HMRCPSP) (Kosztyán and Szalkai, 2020), which is a combination of the traditional and agile algorithms (see the results of an example of HPMa in Figure 3.1), is referred to as the *hybrid project management agent* (HPMa).

This study compares the success (i.e., extended sense of feasibility) of different kinds of project management approaches on different kinds of real project structures and various simulated risk factors. A novel matrix-based risk assessment tool is also proposed.

3.3 From Monte Carlo simulations to survival analysis

Monte Carlo simulation (MCS) is one of the most frequently applied methods of risk management. This is a useful technique to simulate project risks and uncertainties. In MCS, risk effects, such as delays, cost overruns, and overwork, can be simulated by changing the time/cost/resource demands of the tasks (Kwak and Ingall, 2007). In MCS, task demands follow theoretical or empirical distributions. By combining MCS with matrix-based techniques, the interdependencies of the risks can also be modeled (see Section 2.6). In the case of flexible project structures, the project can be restructured (Kosztyán and Szalkai, 2018; Kosztyán and Szalkai, 2020), which until now has received little attention in the literature, but this extension is crucial for handling flexibility, such as in agile and hybrid projects.

Survival analysis is a branch of statistics for analyzing the expected duration of time until one or more events happen, such as failure in mechanical systems, or in this case, project failure. In this study, the survival analysis attempts to answer questions such as the following: what is the population proportion of a project plan that can be managed, and which ones will fail? Can multiple causes of failure be taken into account? How do particular circumstances or characteristics increase or decrease the probability of survival? The main focus of survival analysis is on time-to-event data. Nevertheless, similar to the time-to-event data, the stratification of risk factors can also be modeled.

Typically, survival data are not fully observed but rather censored. Due to the presence of the censoring in survival data, the standard evaluation metrics for regression, such as the root mean squared error and R^2, are not suitable for measuring the performance in random forest-based survival analysis (Ishwaran et al., 2011).

Survival data are commonly analyzed using methods that rely on restrictive assumptions such as proportional hazards. Further, because these methods are often parametric, nonlinear effects of variables must be modeled by transformations or expanding the design matrix to include specialized basis functions. Since following a meta-network analysis means that the analyzed risk factors can be related to each other arbitrarily, a robust flexible method, Ishwaran et al. (2011)'s method, namely, "survival random forest", is applied.

The main advantage of the random forest-based survival analysis (RFS) method is its robustness, such as indicated in its handling of the correlation and dependency between the risk factors and the flexibility it affords for being combined with meta-network analysis.

3.4 Project plan databases

The first problem was to select adequate project plans from a project database because neither known project generators (such as ProGen (Kolisch and Sprecher, 1997a), RanGen I (Demeulemeester, Vanhoucke, et al., 2003), and II (M. Vanhoucke et al., 2008)) nor open project data sources (such as MMLIB (Peteghem and Vanhoucke, 2014) and PSPLIB (Kolisch and Sprecher, 1997a)) distinguish mandatory and supplementary tasks or consider strict and flexible dependencies. Therefore, there are no score values linked to task completion or task dependencies. Nevertheless, without considering flexible dependencies and priorities of task completion, the flexible project plans cannot be modeled because lower-priority (supplementary) tasks cannot be postponed, and the project plan cannot be restructured. Since there is still no real project database that contains an empirical distribution of the priorities or the flexible dependencies, the selection of tasks/dependencies and priorities followed a uniform distribution.

Project databases are essential in facilitating research on various scheduling and resource allocation methods. They enable the comparison of existing methods and the creation of new approaches (Brucker et al., 1999; Hartmann and Briskorn, 2021). In the literature, three categories of data sources are commonly observed: notional data, artificial data generated for research purposes, and empirical data collected from real-world sources.

Single project data are available from various databases, such as

- Patterson (Patterson, 1976)
- Boctor (Boctor, 1993)
- SMCP and SMFF (Kolisch, Sprecher, and Drexl, 1995)
- PSPLIB (Kolisch and Sprecher, 1997b)
- RG300 and RG30 (Debels and Vanhoucke, 2007) (Vanhoucke et al., 2008)
- MMLIB (Peteghem and Vanhoucke, 2014)

to support simulation and evaluation works.

The real-life project database by J. Batselier and M. Vanhoucke (2015) or sets of individual or multiple projects such as

- MPSPLIB (Homberger, 2007)
- BY (Browning and Yassine, 2010)
- RCMPSPLIB (Vázquez et al., 2015)
- MPLIB (Eynde and Vanhoucke, 2020)

also enabling comparative work for researchers as all the above databases contain activities and their dependencies and renewable resources.

There are though also some shortcomings for the generated or simulated databases, as most databases do not include costs, quality, or nonrenewable resources, or only two datasets consider structural flexibility with alternative subgraphs, the RCPSP-PS dataset (Kellenbrink and Helber, 2015) and ASLIB dataset (T. Servranckx and M. Vanhoucke, 2019). Also, a limited number of databases have only one completion mode (Patterson, SMCP and SMFF, PSPLIB, RG300, and RG30), whilst others have multiple completion modes (PSPLIB, Boctor, and MMLIB). Peteghem and Vanhoucke (2014) were highlighting also challenges for the databases related to the low diversity in the complexity of topology networks indicated by the order strength values or further issues that some instances are infeasible. As the target of this thesis is to utilize usable databases for proof of concept, the current databases are satisfying. Space for further analysis and improvement will be part of future work.

3.5 Data sources

Continuing the line of thought from the previous section, the second problem is that the quality parameters are neglected and the cost parameters are also usually missing from the project plans. Nevertheless, these project databases and project generators have been validated and applied in several publications for testing and comparing algorithms; therefore, in this study, it was decided to use the logic network and resource demands, and the project plans have been extended with cost, quality and score parameters in the simulation. The costs are considered as the cost of resources; therefore, they are calculated as follows:

$$c_{i,w} = t_{i,w} \cdot \mathcal{C} \sum_{\rho} r_{i,\rho,w} \tag{3.1}$$

where $c_{i,w}$ is the (resource) cost of task i completed by mode w, and $r_{i,\rho,w}$ is the resource demand for resource ρ of the task i with completion w. The $\mathcal{C}$ is the specified unit cost (e.g., EUR / hour). In the simulation, $\mathcal{C}$ is specified as 1.

When calculating quality, the Babu and Suresh (1996)'s cost-quality trade-off formula is used.

$$q_{i,w} = c_{i,w} / c_i^{\max} \tag{3.2}$$

When the cost is maximal, the relative quality is 1; however, a lower cost provides lower quality. According to Kosztyán and Szalkai (2020) the (relative) total project quality (TPQ) is the ratio of the sum of quality parameters of implemented tasks per the sum of maximal quality parameters of all tasks. This value is maximal if all tasks are implemented in the best quality way. However, this value decreases if either a task is ignored/postponed or even implemented but with lower quality.

These formulas were only required when cost demands and quality parameters are generated for the tasks; however, these values can be modified in the phase of the simulation.

3.5.1 Selection and simulation criteria for initial projects

The aim of the selection and generation of initial project plans is to meet as much as possible the expectations for (IT) software project plans, especially the features of agile and hybrid projects:

CR_1 *Criterion of project structure*: In previous studies,Tavares et al. (1999) and Vanhoucke (2012) showed that software projects usually contain more parallel tasks; therefore, according to Tavares et al. (1999) and Vanhoucke (2012), the number of parallel tasks is greater than the number of serial tasks[1]. Nevertheless, several agile methods, such as the KANBAN and SCRUMBAN methods, limit the number of parallel work-in-progress (WIP) tasks and allow only 3-5 WIP tasks. Therefore, in the simulation, the number of WIP tasks must be lower than 5.

CR_2 *Criterion of task numbers*: Projects are usually separated into smaller autonomous subprojects (sprints) (see, e.g., Dingsøyr et al., 2012) that should be completed within 2-5 weeks; therefore, the number of tasks is limited and should not be greater than 50.

CR_3 *Criterion of resources*: It contains at least two types of renewable resources (e.g., programmer and tester)

CR_4 *Criterion of completion modes*: It contains three completion modes to apply MR-CPSP, and in this manner, it also tests the performance of the hybrid approaches.

The abovementioned criteria were true only for the simulated IT projects. Nevertheless, control group project plans, whose characteristics are closer to construction projects or traditional waterfall software development projects, are also included. Three kinds of datasets were selected. The logic networks, i.e., tasks and their dependencies, of Dataset A are from standard project databases. Project plans of Dataset B are generated by the standard project generator software ProGen, and project plans of Dataset C are from a project database containing real-life project plans.

Logic plans and resource demands are left untouched; however, for the cost and quality domains, formulas (3.1)-(3.2) are also used to calculate the initial cost and quality parameters.

Dataset A contains selected data from the project databases

PSPLIB (j30 dataset) and MMLIB (MMLIB50 dataset). Database selection was performed based on the specified criteria (CR_1)-(CR_4), including the number of activities and serial/parallel indicators, of which the values best fit the projects in the IT sector. To select the appropriate data instances, the average values of several project network topology indicators were calculated [2] of both real-life IT projects and Construction projects (also found in Dataset C). Then, the same set of indicators for the instances of PSPLIB's "j30" and "MMLIB50" datasets were calculated. By minimizing the standard deviation between the results, it was possible to filter the artificial project instances that were closest to the projects in the IT and construction sectors. Ten logic plans have satisfied the above-specified criteria the most. Since project duration and allocated resources over time depend mainly on the structural parameter i_2 (see, e.g., Alfieri et al., 2012; Burgelman and Vanhoucke, 2019), for the control group,

[1] Following the simulations of Tavares et al. (1999), $i_2 = (m-1)/(n-1) \in [0.2, 0.3]$, where m is the number stages in a topological ordered network and n is the number of tasks. $i_2 = 1$ if all tasks are completed in a serial manner, and $i_2 = 0$ if all tasks are completed in parallel.

[2] such as $i_2, i_3, i_4, i_5, i_6, OS, CNC$; for definitions, please refer to M. Vanhoucke et al. (2008)

project plans were selected with the indicator $i_2 \sim 0.4$, which is more specific to the construction projects.

The proposed Dataset A contained the following: (for specifying IT projects: j3031_7; j3035_10; j3042_1; j3031_5; j3064_10; J5063_4; J5046_2; J5043_5; J5050_1; J5061_1; for specifying construction and waterfall projects: j3028_8; j3031_5; j3031_7; j3035_10; j3042_1; j50101_3; j5073_4; j5087_5; j3089_1; j3089_5).

The project plans in groups 1-10 emulated the IT projects, where $i_2 = 0.2$, whereas the control groups (11-20) included a selection of 10 additional projects that emulated the construction project or the traditional waterfall software development projects. The groups 1-5 and 11-15 project plans contained 30 tasks, while those of groups 6-10 and 16-20 had project plans that included 50 tasks.

This database contained 3 completion modes and two kinds of renewable resources.

Dataset B In addition to the selected instances from existing standard datasets, project instance generators have been considered as another source of project data. The widely accepted generator ProGen (Kolisch and Sprecher, 1997a) was selected for this work because it allows the generation of project data with multiple execution modes and supports a wide range of controllable problem parameters (Demeulemeester, 1996; Demeulemeester, Vanhoucke, et al., 2003; Walter, 1995). Ten project structures were generated regarding the criteria (CR_1)-(CR_4), where i_2 was 0.2. Ten projects for the control group were also generated, where i_2 was 0.4. Half of the generated projects have 30 tasks; the other half of the projects contain 50 tasks.

Both the project generator ProGen and the project dataset MMLIB contain only the duration and resource demands of the completion modes; cost and quality are always missing, and because the main cost of the IT project is the cost of resources, the quality parameters are estimated by using formulas (3.1)-(3.2).

Dataset C consists of empirical project data from the database presented by Batselier and Vanhoucke (2015)[3]. **IT projects include the following:** C2011-05 Telecom System Agnes; C2011-07 Patient Transport System; C2011-09 Commercial IT Project; C2012-01 Manufacturing Tool Cost Module; and C2012-09 Digipolis Talent Management Suite. **For control groups, the projects include the following:** C2011-08 Sports Center Tielt; C2011-10 Building a House; C2012-02 Nut Mixing Station; C2012-14 Sluiskil Tunnel; and C2012-17 Building a Dream.

The considered IT and construction projects contained time, cost and resource demands but did not contain completion modes. Therefore, to compare the project management approaches, other completion modes were generated. The original demands, and the generated demands ($d_{i,w}$) for task i were considered and completion w were approximately the original demand (d_i). Formally: $d_{i,w} \in [0.8 \cdot d_i, 1.2 \cdot d_i]$.

Since quality parameters are missing from every known project database, they have to be calculated according to the quality-cost trade-off functions (see Eq. (3.2)). After selecting project plans (see the 20 selected project structures in Dataset A and the 10 selected project structures in Dataset C) and generating 20 project plans in

[3]database url: `http://www.projectmanagement.ugent.be/research/data/realdata`

Dataset B, the original database contained 50 project plans. Half of them were considered an IT project, and half of them were in the control group. In terms of project planning, the main difference between the selected project group and the control group was the project structure. Nevertheless, the distinction between mandatory and supplementary tasks and the distinction between the fixed and flexible dependencies between tasks are also missing from the original datasets. Therefore, the flexibility parameter ($F\%$) is set to be 0%, 10%, 20%,.., 50%, which means that the $F\%$ of task completion and task dependencies is selected to be flexible. Score values, which reside in the interval [0,1], are linked to them. The final database had $50 \times 6 = 300$ PDM matrices.

3.5.2 The proposed meta-network structure and the stages of risk simulation

The proposed meta-network structure is a (meta)model for project risk management. It has four parts: stakeholders, risk factors, risk effects, and goals. In this framework, three groups of stakeholders are specified: the customers, who order the software; the management, who manage the progress of the project; and the developers, who make the software. Risk factors address the change of constraints in the contract phase (stage one), the change of demands in the scheduling phase (stage two), and those in the project tracking phase (stage three). These risk factors may influence all the risk effects, such as the delay of the project duration (ΔTPT), the overbudget situation (ΔTPC), the changes in resource demands (ΔTPR), the changes in project quality (ΔTPQ) and the changes in the project scope, which is quantified by the total project score (ΔTPS). The stakeholders may have different goals that are partly or fully contradictory to each other. Usually, customers want the highest quality software ($TPQ \rightarrow$ max) with considerable functionality ($TPS \rightarrow$ max), but as soon as possible ($TPT \rightarrow$ min). Management tries to minimize the budget ($TPC \rightarrow$ min) and similar to the developers, they try to decrease the use of resources ($TPR \rightarrow$ min) as much as possible (see Figure 3.2). Nevertheless, customers are usually not interested in decreasing the project cost or reallocating resources while the project budget can be maintained.

The proposed simulation framework has three stages. In *stage-one* constraints, such as the time ($C_t\%$), cost ($C_c\%$), quality ($C_q\%$), score ($C_s\%$) and resource ($C_r\%$) constraints, are the results of the contract; therefore, in this stage, by an agreement with the customer, there is an opportunity to alter the constraints. According to the specified goals, managers can select the adequate project management approach, which is represented as an agent. An agent tries to produce feasible project plans. In addition to the feasibility, the scheduling properties, such as scheduling performance (project duration, project cost, and resource demands per adequate constraints), are also explored. ***In stage one the contractual stage), the emergence of bargaining between customers and developers is modeled.*** More restrictions can produce fewer completed tasks and lower quality but can produce a lower budget and lower project duration. More requirements can produce more completed tasks but can produce a greater budget and greater project duration. Knowing the priority and completion mode data, the minimal and maximal value of the total project cost (TPC), the total project time (TPT), the maximal value of total project resources (TPR), the total project qualities (TPQ), and the total project scores (TPS) can be specified following Kosztyán and Szalkai (2018). Constraints are the subject of bargaining (see the rate of constraints $c_x\% \in [0,1]$, where $c_x\% = (c_x - \mathrm{TPX}_{\min})/(\mathrm{TPX}_{\max} - \mathrm{TPX}_{\min})$,

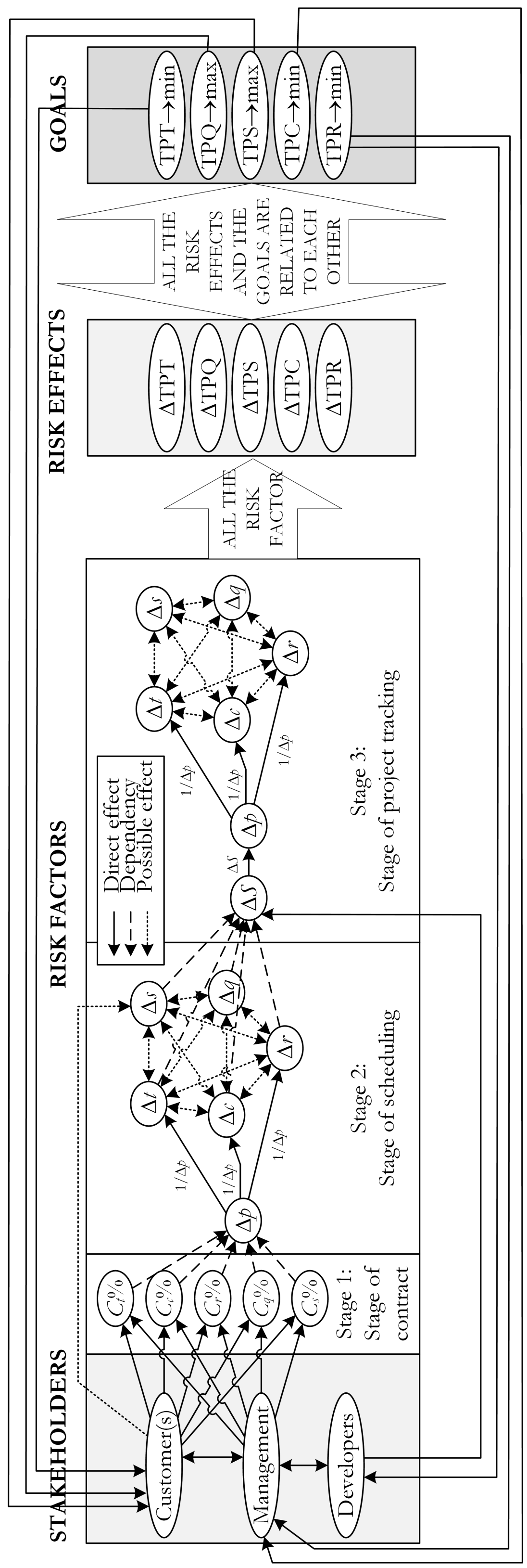


FIGURE 3.2: The proposed meta-network structure

$c_x \in [\text{TPX}_{\text{min}}, \text{TPX}_{\text{max}}]$ is the time/cost/resource/score or quality constraint). Furthermore, TPX can be TPT, TPC, TPS, TPR or TPQ.

The target functions are either $TPT \rightarrow$ min or $TPC \rightarrow$ min or $\overline{TPR} \rightarrow$ min or $TPQ \rightarrow$ max or $TPS \rightarrow$ max, $\overline{TPR} = mean(TPR) = \frac{1}{r}\sum_{\rho=1}^{r} TPR_{\rho}$. At the end of this stage, a set of feasible project plans managed by TPMa / APMa / HPMa is specified.

In *stage two* (the scheduling stage), only the feasible (i.e., survived) project plans are considered (see dashed lines between constraints in stage one and the project properties in stage two in Figure 3.2). At this stage, the time/cost/resource/quality and score demands are varied a.) independently and b.) considering interdependencies modeled by the matrix representation of MNA. Due to the MNA, the extreme or shock effects can also be modeled. Shock effects are limited in range and in the number of affected tasks but have a higher impact on these. In the planning phase, such an effect could be the replacement of a vendor, who delivered software or hardware solutions to some of the tasks, and the new vendor has different costs, delivery time, or different resources that are required to implement its product into the project. In the tracking phase, a virus infection or system shutdown could be typical examples, where only those tasks are affected that are in progress when the event occurs. In this simulation, a two-step Monte Carlo analysis is used, where the set of tasks that will be modified are specified first. The selection of tasks was random. In this simulation, this selection parameter is specified as $\Delta p = 10\%$ and $\Delta p = 100\%$. When all tasks are modified ($\Delta p = 100\%$), the uncertainty of planning is analyzed; when only 10% of tasks are affected ($\Delta p = 10\%$), the *shock or extreme effects* can be modeled. In the latter case, according to the literature (see, e.g., Zafar et al., 2018), the modification of task durations (Δt), cost demands (Δc), and resource demands (Δr) will be 5-10 times larger than the effect of the uncertainty. To avoid the overemphasis of the shock effect in the comparison, in the simulation, the impact of the shock is inversely proportional to the affected range (Δp) (see Figure 3.2). This means the impact is 10 times that of the (beta-distributed) variation that models the uncertainty, but it concerns just 1/10 of the tasks, so a more focused effect is compared to a more distributed effect, while the cumulative effects in the two cases are commensurable. In the case of $\Delta p = 100\%$, can be focused on the estimation uncertainty, where every task demand can be uncertain; for $\Delta p = 10\%$, it was concentrated more on the risks of implementation, where not all demands are varied, but this variation can be much greater than the uncertainty of the estimations.

3.5.3 Sensitivity

A novel element in the proposed framework is the sensitivity analysis of the task priorities (Δs). Currently, the use of the conditional risk factor (Δp) is very rarely used in simulations; nevertheless, based on the author's knowledge, none of the risk management methods model the varying of the customers' requirements and priorities as the proposed framework does. The varying priorities are specified by changes in diagonal values in the logic domain of PDM. The off-diagonals are specified by the task completions' probability according to the varying customer requirements. Following the practice of sensitivity analysis in project management, all changes in parameters, such as time, cost, and resource demands as well as the quality and score parameters, follow a β-distribution, where the most likely value was the original value of the parameters, the optimistic value was 90% and the pessimistic value was 130% of the original value. This set of parameters follows the underestimated demands observed in practical life.

Stage one and stage two simulate only the modifications of project plans according to the varying customer requirements.

In *stage three* (the tracking stage), the process of the implementation is simulated. Here, completed tasks are not varied; however, work-in-progress tasks and unstated task demands and priorities are varied.

3.6 Implementation of the simulation framework

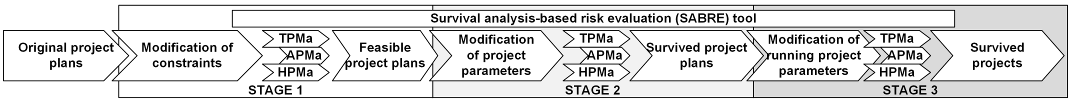


FIGURE 3.3: The proposed simulation framework

Figure 3.3 shows the proposed simulation framework. In this simulation, the influence of risk effects, such as the modification of constraints (see stage one) and overruns of cost and time (stage two and stage three) are mitigated by project management agents. The properties of the survived projects handled by different kinds of project management agents and their count are compared for all of the five specified goals.

3.6.1 Stage one - the stage of project contract

At stage one, time/cost/quality/resources and score constraints are set to be $\frac{1}{3}$ or $\frac{2}{3}$ of the theoretically available range of the project demands. These parameters simulate two deals. One of the deals is more restricted, the other one is more relaxed. In this way, it could be obtained 2^5 (*number of possible constraint sets*) $\times$ 50 (*number of projects*) $\times$ 6 (*levels of flexibilities*) $= 9,600$ problems. For all of the five specified target functions, can be obtained $9,600 \times 5 = 48,000$ scheduling problems. These problems are solved by TPMa, APMa, and HPMa agents. Therefore, it can be gained gain $3 \times 48,000 = 144,000$ results. The results solved by agents are compared by their rate of feasibility (feasible projects/all projects) and by their scheduling performance (see Eq. (3.3)).

$$TPX\% = \begin{cases} \frac{c_x - TPX}{c_x - TPX_{\min}}, & \text{if} \quad TPX \in \{TPT, TPC, TPR_\rho\}, \rho = 1, 2, .., r \\ \frac{TPX - c_x}{TPX_{\max} - c_x}, & \text{if} \quad TPX \in \{TPQ, TPS\} \end{cases} \tag{3.3}$$

where $c_x \in \{c_t, c_c, c_{r_\rho}, c_q, c_s\}$.

Regarding the $TPX\% \in [0, 1]$, the greater value indicates better performance. If $TPX\% = 1$, it means that when optimizing, the best value (such as the minimal project duration, minimal project cost, minimal resource demands, maximal project quality, or maximal project score) can be reached, whereas if $TPX\% = 0$ exists, only the constraint can be satisfied.

3.6.2 Stage two - the stage of project scheduling

Since infeasible project plans do not assume to pertain to any concluded agreement, at stage two (stage of scheduling), only the feasible solutions are surveyed. Two scenarios are explored: (1 - sensitivity analysis of uncertainty, i.e., $\Delta p = 1.0 = 100\%$) applies to the scenario when all parameters, such as time/cost/resource demands and score/quality parameters of tasks can be changed between -10% and 30% and parameters follow the three parameters (a, m, b) of the β-distribution, which is usually used in Program Evaluation and Review Technique (PERT) networks. The most likely values (mode, m) of the parameters in this distribution are the task time/cost/resource demands, which are specified in stage one. $a := 0.9m, b := 1.3m$; (2 - sensitivity to shock effects) applies to scenarios when only parameters of randomly selected tasks ($\Delta p = 0.1 = 10\%$) are changed, but these changes are $1/\Delta p = 10$ times of the uncertainty effect. Although the applied survival random forest method is not sensitive to the correlation between the risk factors, in order to explore the influence of correlation between risk factors (i.e., Δt, Δc, Δr, Δq, and Δs) to risk effects (such as ΔTPT, ΔTPC, $\Delta \overline{TPR}$, ΔTPQ, and ΔTPS), a subgroup, where the mean correlation between risk factors is greater than 0.6, is also specified and explored. The ratios of changes in project parameters are calculated as follows:

$$\Delta TPX_{i,j}\% = \frac{TPX_i}{TPX_j} \tag{3.4}$$

where $i = 2, 3; j = 1, 2$ is the number of stages. $TPX \in \{TPT, TPC, \overline{TPR}, TPQ, TPS\}$.

For example, $\Delta TPT_{i,j}\% = 1$ or $\Delta TPC_{i,j}\% = 1$,.., $\Delta TPS_{i,j} = 1$ and means that due to the applied project scheduling (and in this case, risk mitigation) approach, the total project time/cost/resource/quality/scores are not changed with the changes in the risk factors. If $i > j$, then $\Delta TPT_{i,j}\% \geq 1$, $\Delta TPC_{i,j}\% \geq 1$, $\Delta \overline{TPR_{i,j}}\% \geq 1$ and $\Delta TPQ_{i,j}\% \leq 1$, $\Delta TPS_{i,j}\% \leq 1$ can be assumed. The risk mitigation performance of the project management approach is better if this ratio is closer to 1.

The changes in the feasibility rate are also calculated as follows:

$$\Delta f_{i,j}\% = \frac{f_i\%}{f_j\%} \tag{3.5}$$

where $f_i\%$ is the feasibility rate in stage i.

Similar to $\Delta TPQ_{i,j}\%$ and $\Delta TPS_{i,j}\%$, $\Delta f_{i,j}\% \leq 1$

3.6.3 Stage three - the stage of project tracking

Stage three is based on the result of stage two. In stage three, all risk factors are used that are introduced in stage two. However, in this case, the rate of scheduled tasks ($S\% \in 0.25, 0.50, 0.75$) influences how many tasks are completed or are in progress. In stage three, only the remaining task parameters can be changed, and the agents have to mitigate the risk effects to keep the deadlines and the budget while minimizing the project duration or the project cost or maximizing the quality of the project.

Due to the different natures of the project management agents, it is assumed that the counts and the schedules of the surviving project will be significantly different at the end.

Figure 3.4 shows the operation of the stages of the SABRE via an illustrative miniature project. This project contains only one mandatory (A) and one supplementary (B) task, with two completion modes and two resources. Figure 3.4 shows only the part of the simulation stages, where the applied agent is the HPMa, the applied target function is to minimize TPT and there is no correlation between the changes in the task parameters (risk effects). At Stage 1, different kinds of constraints are specified to simulate the negotiation (e.g., cost, deadline) between the vendor and the customer. Figure 3.4 shows that if the HPMa cannot comply with all restrictions,

Stage	Risk factor/source	Notation
Contractual stage (Stage 1)	Constraint strictness / more restrictive requirements	$C_t\%, C_c\%, C_q\%, C_s\%, C_r\%$
Scheduling stage (Stage 2) and	Task demand uncertainty regarding the whole project	$\Delta t, \Delta c, \Delta q, \Delta s, \Delta r$
Tracking stage (Stage 3)	Shock-like, high degree changes concerning a narrow set ($\Delta p = 10\%$) of the tasks	$\Delta p \in \{0.1, 1.0\}$, $1/\Delta p \cdot \Delta t$, $1/\Delta p \cdot \Delta c$, $1/\Delta p \cdot \Delta q$, $1/\Delta p \cdot \Delta s$, $1/\Delta p \cdot \Delta r$
Tracking stage (Stage 3)	Rate of completeness: Already completed tasks reduce the adaptability of the management approach	$S\% = \{25\%, 50\%, 75\%\}$

TABLE 3.3: Sum of risk factors and risk sources

in such cases, the contracting process miscarries and the vendor cannot undertake to complete the project with the original customer specifications.

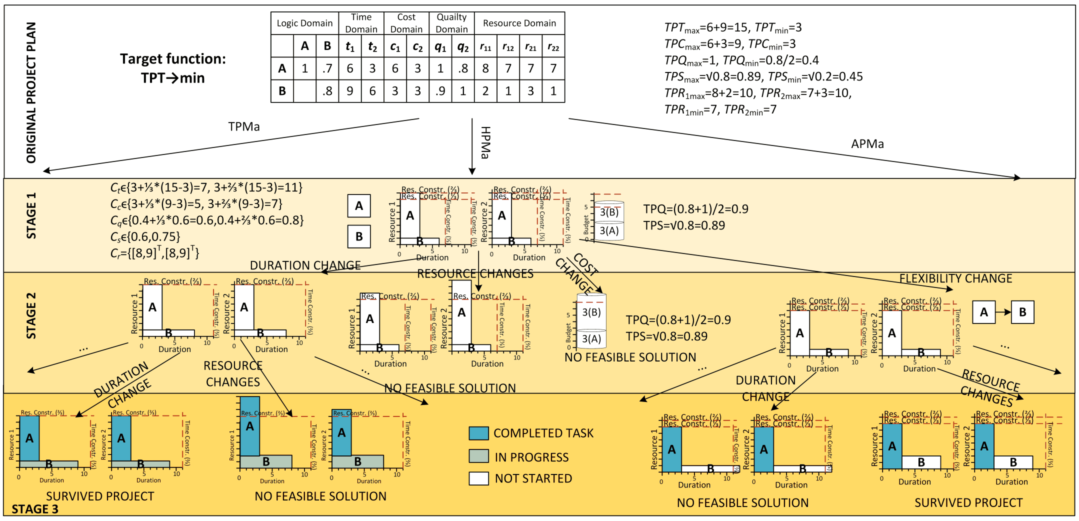


FIGURE 3.4: Example of simulation process in SABRE

In Stage 2, the deviations of activity resources or time demands from the planned values are analyzed as risk factors. If a project plan cannot be implemented with the chosen management approach without renegotiation of specifications, costs, or deadlines, then it has no feasible solution, and it is deemed as a non-survived project. In both survived and non-survived project cases, the causes can be followed alongside the branches. Figure 3.4 shows that the miniature project can be solved only for the relaxed resource constraints (in Stage 1). If the flexibility is not changed in the scheduling stage (Stage 2), the project is more sensitive to resource and cost changes than to duration changes. However, if the flexible dependencies were to become fixed (e.g., because of the technology change), the serial completion would be more sensitive to task duration changes.

Table 3.3 shows the summary of risk factors and risk sources in the phases.

3.7 Applied metaheuristic optimization

The resource-constrained project scheduling problem is a scheduling problem that involves the allocation of tasks in a way that minimizes the makespan. Nevertheless, it has been established that the RCPSP is an NP-hard combinatorial problem. To rephrase, the problem is difficult to solve within an acceptable amount of time using computational methods. Consequently, a multitude of metaheuristics-based methods have been devised to locate solutions that are close to optimal for the RCPSP. Genetic algorithms have been successfully utilized in a diverse range of combinatorial optimization problems, demonstrating their efficacy. Applying thus NP-hard problem solutions with heuristics are accepted in the scientific community to reduce the non-linear solution time to a close to linear approach, and in this case is also the situation. Not looking for the exact and fully optimal case, however, the near-optimal solution can be accepted respecting that its solution time is significantly shorter.

3.8 Plan and design of case study

Case studies are an effective tool for understanding real-world scenarios and applying theoretical concepts to practical situations. Due to the fact that related to application lifecycle management and flexible projects, only a strongly limited amount of theoretical and empirical study materials are available, it was obvious to examine the possibility of leading a case study to contribute to the validation of the research of the thesis and potentially to the academic literature also. The target was to satisfy the objectivity, validity, and generalizability of the selected case as a fundamental demand.

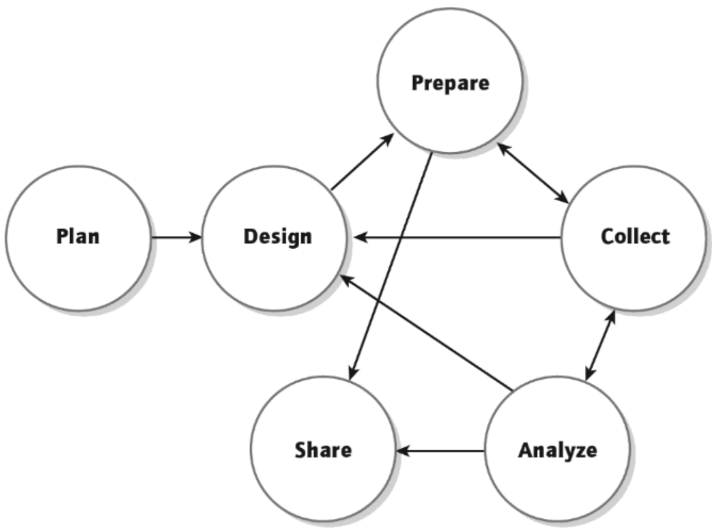


FIGURE 3.5: Case study research: linear but iterative process (Yin, 2009)

Using case study methodology is beneficial for research to reinforce the findings with different aspects in a real environment. Yin (2009) highlights that the case study is a linear but still iterative process, see Figure 3.5. Each step in the linear process of case design (planning, designing, preparing, collecting, analyzing, and sharing) forces the researcher to review and re-examine former decisions. For such a novel research area as ALM this results in a higher confidence and acceptance level.

As case study methodology has long been a contested terrain, despite the fact that it is one of the most frequently used research methodologies, the methodologists do not have a full consensus over the design and implementation of the case study (Yazan, 2015). Therefore, a thorough examination and evaluation were necessary to decide which research school method to adopt for this ALM case study.

A superb summary and comparison from the spectrum of different views and conceptualizations is available by (Yazan, 2015) about Yin, Merriam and Stake, who are the three prominent authors to provide procedures to follow when conducting case study research see in Table 3.4.

(Yin, 2009) for *source* usage pragmatically claims that researchers selecting either qualitative or quantitative research, there is a strong and essential common ground between the two, which is useful for discovering a new area like ALM. Opposingly, Merriam's and Stake's viewpoint is that the case study should focus on qualitative sources only, which in this case would lessen the capability for measurable results. In the case study, the targeted *definition* is to validate the flexible model's extended applicability in the ALM context, "how" it is fitting and "why" can be an improvement. The definition of case study thus is closest to the one by Yin's. A primary distinction in *design*ing case studies is between single- and multiple-case study designs. Yin summarizes and provides a descriptive overview in Figure 3.6 about the typology of the case study in his view. Stake claims that design is rather a flexible approach with some target research questions which during the case study might change also, thus it has a progressive focus during the evaluation. Meriam claims that the design is based heavily in the literature review, from where the framework and research questions must come, and the case study serves as confirmation only on these specific and well-defined questions. For current target, to validate the model for feasibility at the first step it is desirable to see how it is working in a specific but well-defined context. The focus is fixed, thus the Stake approach is out of consideration. As the target is to validate the model, which is novel to literature, Meriam's approach is excluded also, leaving Yin's approaches to be examined. The useful approach for the ALM model application would be to target to the scheduling feasibility and performance in a specific context, if possible with more examples.

From Figure 3.6 this is reflected in Type 2, Single-case embedded design. Worth mentioning, that as a future step to broaden the view for different application areas with comparison options also, the multiple-case design would be preferred. *Data gathering* from the ALM environment is beneficial to have multiple sources. Not only factual data from file versioning systems but also their validation and context information from experts, and managers via interviews, and expert discussions. It is important to understand and well define the context to be able to analyze and conclude the case. Crucial the widest set of data gathered so that adaptation as input for the solver can be also well determined. After the experimental run of the simulation on the gathered *Data analysis* will proceed to compare the simulation results with the results and experiences from the actual run. Expectations are to identify factors that later on in the simulation can be extended for improvement purposes. *Validating Data* are strongly related to the Data gathering, as the most well-defined data availability as input for the simulation influences the analysis result and conclusions. For validation techniques there are several methods, the plan is to use triangulation for sources, i.e. primary sources from the file versioning system, validate it with experts, and review it with related competency managers.

Case study attributes	**Robert Yin's Approach**	**Robert Stake's Approach**	**Sharan Merriam's Approach**	**Proposal for ALM case study**
Research Sources	Quantitative and Qualitative sources to be combined.	Exclusive Qualitative sources	Exclusive Qualitative sources.	Quantitative and Qualitative sources to be combine to maximize the data availability.
Definition	An empirical enquiry addressing the "how" and "why" questions.	A study of the particularity & complexity of a single case. Holistic, Empirical, Interpretive, Empathic.	An intensive, holistic description and analyis of a bounded phenomenon. Particularistic, Descriptive, Heuristic.	Yin's approach the closest, to see"how" the scheduling is handled in real life and "why" the proposed method might have improvement.
Design	A logical sequence that connect the empirical data to research questions and then to results. Four types: single holistic, single embedded, multiple holistice, multiple embedded.	Flexible design, research questions. Progressive focus.	Five steps: Literature review; theoretical framework creation; research problem identification; sharpening research questions; purposive sampling.	One context with multiple subunits possibly could get a good validation view in the case study.
Data Gathering	Documentation, archival records, interviews, direct observations, participant observation and physical artifacts.	Observation, interview and document review.	Interviews, observing, and analyzing documents.	Factual data collection and insights from interviews and observations.
Data Analysis	Five dominant techniques: pattern matching, explanation building, time-series analysis, program logic, models, and cross-case synthesis.	Two strategic ways to analyze data: Categorical Aggregation and Direct Interpretation.	Six analytic strategies: ethnographic analysis, narrative analysis, phenomenological analysis, constant comparative method, content analysis, and analytic induction.	Explanation building for scheduling efficiency.
Validating Data	Construct validity, internal validity, external validity, reliability.	Triangulation for data source, investigator, theory and methodology.	Validity and reliability checks with Qualitative methodologies.	Triangulation for sources, analytic tools usage.

TABLE 3.4: Case study methodology selection decision matrix
(Source: own edit)

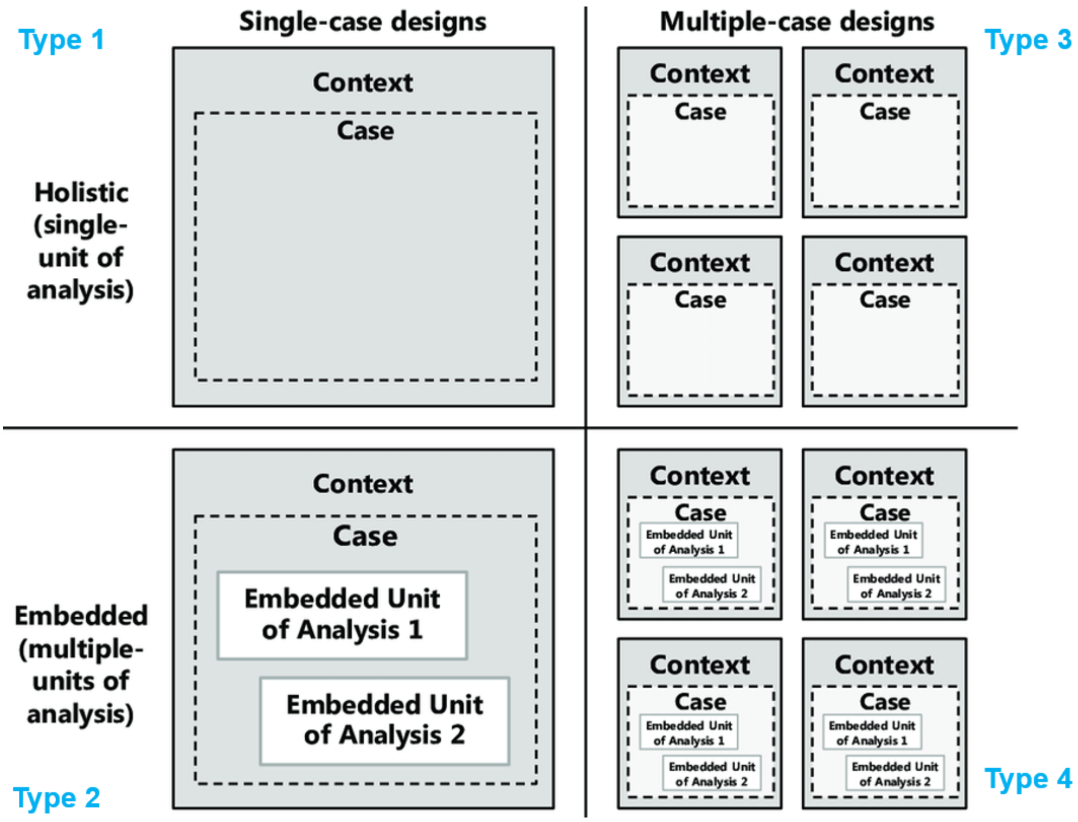


FIGURE 3.6: Case study types by Yin (2009)

As a conclusion, **a single-case embedded design** methodology (Yin, 2009) was chosen to support the ALM case study research. Reflecting in the following stages defined by Yin will be followed for the case study: Plan, Design, Prepare, Collect, Analyse and Share.

The empirical information is imperative for the validation also as the performance is considered in terms of relative values (ratios) (see Eqs. (3.3)-(3.5)). The results show that a risk analysis should also include real-life projects because their constraints may be different from project structures in a standard database.

The preparations phase includes the a priori information collection, identifying stakeholders as primary information sources, databases and tools for secondary sources, and self-preparation for the case.

The case study was carried out at a global automotive supplier established in 1871, a leading company that specializes in manufacturing brake systems, interior electronics, automotive safety, powertrain and chassis components, tires, and various other automotive parts. The organization operates in 58 countries, with a total sales of €33.8 billion and an employee count of approximately 190,000. In this case, the focus is on electronic brake systems' software application, where the company is a top-tier supplier and competes with well-known companies, which showcases the organizational structure and key data.

Recent years have challenged the supplier to a new approach from several vehicle production companies, that the installed brake system during production requires frequent SW updates after retails (Stepanović et al., 2018; Martin, 2023). Not only problem fixes are addressed but also introducing new functions that may be installed and activated during the cars' post-production lifespan. The phenomenon is strongly related to the fast pace of the development time reduction to reach quicker time-to-market ratios thus the hardware is already available at the production time and for the SW the original equipment manufacturers (OEMs) decide to roll out new functionalities in the vehicles in a later time only, which can be days, months, even years. Such functionalities or features also can be purchased and downloaded by the end customers. Usually, this can happen either in a service garage or via wireless methods. For example, one German OEM offers their functionalities via the ConnectedDrive store to their customer for Driving Assistant, Parking Assistant, and Active Cruise Control with Stop and Go functionalities to purchase (Source: *BMW ConnectedDrive* 2023). Important to highlight here that application development is still called application software project development, even though the characteristics are already relevant to application lifecycle management due to the frequent additional scope change, and the scattered and repeated development phases during the elongated lifetime.

The organization unit examined in this case study is part of the Research and Development area, which is responsible for, but not limited to the SW developments during the lifetime. This includes from the start of the planning of the application through the main development cycle till the series release and after that, the post-production SW updates till the end of the lifetime of the vehicle productions by the given vehicle OEM. The R&D organization has a matrix organization overall, following the automotive traditional V-model style development with multilocation development centers, however, in recent years the efficiency of V-model based development has also been questioned in the organizations. Similarly, as B. Liu et al. (2016) points out, the last decades' SW heavy developments put severe challenges to the traditional V-model, which often occurs with very high costs in the late verification stage and elongates the response to the changes from customer to the market, especially in the case of considerably high system complexity. Thus this company in some cases willing to focus and improve the collaboration with the OEMs, so that several times the work is done between partial or fully agile teams and ad-hoc organized for projects. This means the adaptation of Scrum for SW development with biweekly SW delivery and incremental approach, participation in content planning in PI (Product Increment) planning within the customer organization, etc. Even though there is no clear implementation of the SAFe (Scaled Agile Framework enterprise - Knaster (2023)) organization introduced, there is an ongoing investigation for its feasibility.

Overall, the ALM-related attributes are met and this is the reason for the ALM case study was selected for this specific SW application.

As a summary for the Research framework and Methods chapter, the Research Assumptions (RA1-RA3) were presented in related to the Research Questions (RQ1-RQ3). Then the simulation framework was presented related to the methodological research part of the thesis to understand the setup of simulation tool, including the agents representing the scheduling methodologies (traditional, agile, hybrid), the simulation engine, the used databases, the information sources for the project structures. Then the Stages of the simulation was presented with the applied meta-heuristic optimizations. Finally the case study related design and plans were described in this chapter.

Chapter 4

Results

This chapter contains the results of the systematic literature review for the ALM definition research, which contains the evaluations and results for the ALM definition search and classification. After that the results of the simulation for the methodological research of the scheduling can be found with the descriptive results and their discussion included.

4.1 Literature review results

In this section, the outcomes are provided of the performed Systematic Literature Review (SLR) by following the PRISMA procedures outlined earlier. The SLR aimed to discover sources containing definitions of Application Lifecycle Management (ALM). Additionally, it discusses the findings of the Critical Review, which involved examining these definitions.

As of July 2023, the Google Scholar database contained approximately 3230 scholarly publications related to the keyword "application lifecycle management". These publications were screened and refined using filters. The addition of the definition keyword in Filter 1 resulted in a decrease to 2510 sources. Filter 2 involved refining the language to exclusively English by examining the abstracts. This entailed eliminating the non-English entries (720) and eliminating the remaining duplicates (19), resulting in a total of 876 entries. The eligibility check first conducted a classification process to establish the type of sources. Initially, a scope was established to exclude some categories (such as Policies, University non-reviewed materials, other presentations, etc.), and this scope was further improved based on concerns of quality. The hosting journal for the articles was determined using SCIMAGO, while the ranking for conference papers was determined using Qualis. The Articles (79) from a non-ranked journal and the Conference Papers (247) from a non-ranked Conference that were published have been eliminated. In addition, both the Bachelor's theses (29) and Master's theses (29), as well as the Business articles (87), were excluded due to their lack of rigorous academic peer review. A comprehensive examination was conducted on the remaining items to determine if the definition is expressly stated in the source. If the definition was found to be lacking, the source was eliminated (315). As a consequence, the PRISMA procedure yielded a total of 76 entries.

The definitions found in both the Top and Extended Academic sources may be seen in Table 4.1. This table is divided into two main parts: the upper half displays the top academic entries, while the second and third lines show the extended academic entries. The pie charts represent the provided categories, showing the proportion and specific numbers of those that contain the definition (represented by a light-colored slice) and those that do not (represented by a darker-colored slice).

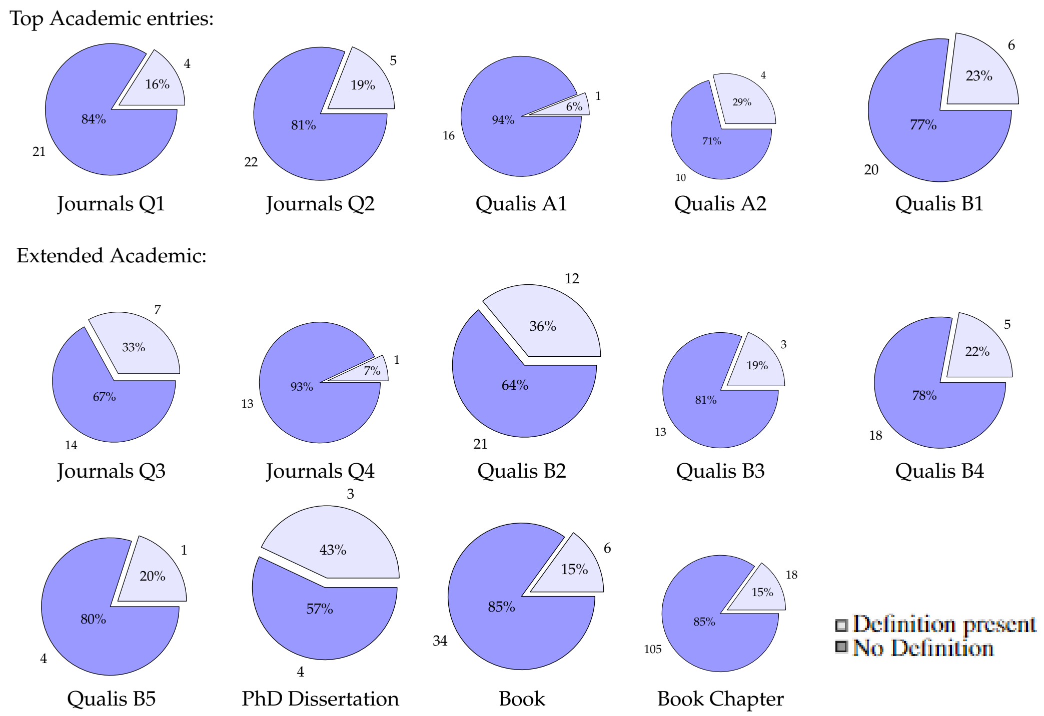


TABLE 4.1: Definition availability in Top and Extended Academic ranking sources

The analysis of the Top Academic entries reveals that the Qualis B1 procedures contain the highest number of definitions (6), while the Qualis A1 proceedings have the lowest number (1). The Articles in Q1 and Q2 have fewer than 20% of the definitions. Within the Extended Academic sources, the definition appeared most frequently in Qualis B2 level publications (12 occurrences), whereas it appeared least frequently in Q4 journal articles (1 occurrence). There were a significant number of explicit definitions in the Q1 articles (7) and Books (6) of the journal.

The objective of the SLR was to gather every relevant resource from which the ALM definition can be obtained. Upon general inspection, the keyword search yields articles primarily including references to single-digit entries. This implies the underdevelopment of the respective field. The search for the keyword "Application Lifecycle Management definition" on the Google Scholar search engine yielded a limited number of literature review articles related to ALM. These articles were not explicitly labeled as review articles in Google Scholar but were identified and included in the final selection of 76 sources after screening and eligibility assessment.

After thoroughly reading the sources, the ALM definition led to a significant reduction in the number of items compared to the meticulous filtering applied in the previous steps. The current definition of ALM can be found in Table 4.1 for the Top Academic ranking and the Extended Academic rankings. Out of the 22 years reviewed, just 20 items made it to the top rankings, which accounts for a mere 18% of the filtered results. The comprehensive academic literature has a total of 282 sources, out of which 56 sources provide a definition. This accounts for approximately 20% of the total.

In summary, a total of 76 sources are given that include the definition of ALM.

4.1.1 Critical Review for ALM definition

The purpose of the critical review is to furnish information about the research questions RQ1 and RQ2. The qualitative analysis of the articles was undertaken after applying filters to the findings. The article gathered the major characteristics, recommended methodologies, and targeted audience for each entry in the ALM definition.

Definitions extraction Evaluation of the identified entries and the explicit search for the definition was proceeded by a full text reading the case by case.

Upon initial examination, it is evident that there are recurring references to the existing literature. However, there are also definitions that present a distinct interpretation of the scope of ALM. Consequently, a compilation of the sources and their ALM definition was undertaken and allocated. If the forthcoming source presented a notably distinct ALM definition, it was then appended to the list of definitions. Significantly different refers to variations either in the comprehension of the extent or the idea. After thoroughly examining all the sources, an assignment was made for each source to a corresponding definition. These assignments were subsequently reviewed and their accuracy was verified.

Table 4.2 contains a summary, the first column describes the scope for the definitions as meta-information, and the second column contains the definition explicitly available in the source, and led to this scope definition.

In Table 4.2 definition A) refers to ALM as *the equivalent of Product Life-cycle Management in the SW domain*. The life cycle encompasses the entire duration of an entity's existence, from its inception to its end, and includes all the activities, tools, and parties involved.

Definition B) refers to ALM *as lifecycle management for SW development and maintenance included*. The scope of this matter is comparable, however, the specifics of life-cycle management are not thoroughly explained.

Definition C) refers to *ALM as a framework for the coordination of activities* (including requirements, modeling, development, build and testing) *and artefact management* (enforcement of the processes for interconnecting activities; management of relationships and links between the development artefacts; and reporting on the progress of the development) during the SW lifecycle.

Definition D) refers to ALM as an extended SDLC (Software Development Life Cyle) with stages after development also. However, not detailing this stage, just referring to it as an extension.

Definition E) refers to ALM *as a paradigm, that contains governance, development, operation/ maintenance*. This enhances the level of abstraction in ALM, serving as a comprehensive integration and management summary. It is a strategy used for the development, operations, and maintenance of software products. Encompasses the entire lifecycle, including the conception, implementation, and termination phases.

Definition F) refers to ALM as *a lifecycle-oriented control of the problem resolution process with the scope only post development services*.

Definition G) refers to ALM as a *process for keeping track of their quality goals*.

Scope. ALM is...	**Definition**
A) a process for SW PLM/SDLC	Product life-cycle Management (PLM) and its equivalent in software, namely application life-cycle management (ALM), is the overall business process that governs a product or service from its inception to the end of its life in order to achieve the best possible value for the business of the enterprise and its customers and partners. PLM/ALM combines processes, people, and tools for the effective engineering of products from their inception until the end of service. It involves tacit knowledge of experts and explicit knowledge, codified in procedures, processes, and tools. PLM/ALM stretches from know-how to know-what and know-why. (Ebert, 2013; Gatrell, 2016; Lacheiner and Ramler, 2011)
B) SW development AND maintenance.	Application Lifecycle Management (ALM), a widely-used lifecycle for software development and maintenance (Rossberg, 2014; Ramler et al., 2012).
C) Artefact management tool for SDLC.	ALM "has emerged to indicate the coordination of activities and the management of artefacts (e.g., requirements, source code, test cases) during the software product's lifecycle" (Kääriäinen, Eskeli, et al., 2009; Gatrell, 2016) The coordination of development lifecycle activities, including requirements, modeling, development, build and testing, through: 1. enforcement of the processes that interconnect these activities; 2. management of relationships and links between the development artefacts used or generated by these activities; and 3. reporting on progress of the development effort as a whole. ALM is often seen as a framework that aims at synchronising all the lifecycle activities instead of focusing on any specific lifecycle activity" (Schwaber et al., 2006)
D) an SDLC extended with phases after development.	ALM is the product lifecycle management of computer programs that is a wider approach than the SDLC, which is limited to the phases of the typical software development stages. In contrast, ALM defines stages after the development lifecycle as well (Government Commerce, 2007; Arya et al., 2011b; Chappell et al., 2010).
E) a paradigm: governance, development, operation/maintenance.	Application Lifecycle Management (ALM) is a recent paradigm for integrating and managing the various activities related to the governance, development, and maintenance of software products. ALM as a combination of three functions: governance, development and operations, and three milestones: (start of) ideation, deployment and end-of-life. (Chappell et al., 2010; Rossberg, 2014)
F) ALM is a service for after development part only	Application management refers to the lifecycle-oriented control of the problem resolution process for operational application systems excluding any fundamental application development services (Arya et al., 2011a).
G) ALM for quality assurance	Establishing a standardized development-to-release workflow, often referred to as the ALM process, is particularly critical for organizations in their efforts to meet tough IT compliance mandates. (Tracy, 2006)

TABLE 4.2: Summary table of ALM definitions and their scopes

In Appendix A.1 can be seen all the assignments between the processed sources and their linkage to the definitions.

Also, the summary for the definition summary for the included literature can be seen in Figure 4.1. The Y axis describes the amounts, the columns represent the definitions A to G like in Table 4.2. The columns are composite from Top Academic literature references (blue color with a number) and the Extended academic (red with a number). It is visible, that there are 3 prevailing definitions: A, C, E.

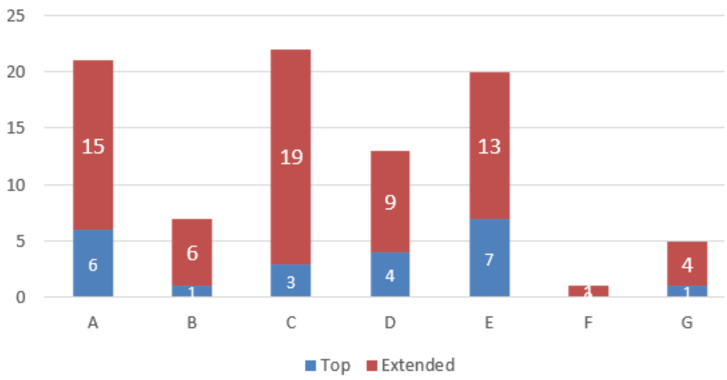


FIGURE 4.1: Summary of various definition scoped present in included entries

The literature research conducted to find the definition of ALM utilized the PRISMA approach, which integrates quality and systematic review into the process. Please refer to Figure 2.13 for a visual representation of the PRISMA flowchart. Firstly, an analysis of the ALM field in academic scope reveals that it is relatively new and limited, as seen by the lower number of records found in the Google Scholar engine compared to the higher number of results in the broader Google search, which includes non-academic content. Nevertheless, the presence of several academic sources suggests that the ALM field is extensively covered in scientific literature and actively studied across all levels and platforms within the scientific community. This suggests that the academics were increasingly becoming interested in the business-related papers provided by the suppliers.

During the filtering process, only entries written in the English language were kept, while sources that potentially contained non-English content were eliminated. This was achieved by including only those entries with English abstracts or translations provided by Google for the search. This method was in line to thoroughly digest the sources while the author was actively involved. Although it is possible that sources of information from global activities were lost, it was seen that high-quality materials were mainly published in English-language journals and conferences. This prompted us to choose them for further study.

The emergence of ALM can be attributed to the inadequacy of companion areas like Product Lifecycle Management (PLM) and Software Development Lifecycle (SDLC) in handling the growing complexity of software application management in the economic environment. The primary benefit of PLM is its comprehensive tracking and overview of a product throughout its lifecycle. However, its limitation lies in its emphasis on the traditional product perspective, disregarding the distinct differences between software and hardware products as they evolve over time. On the contrary, the SDLC offers the benefit of software-specific procedures, but it suffers from the drawback of not taking into account broader lifecycle factors. ALM aims to encompass the entire software development and maintenance process, as well as all associated management activities. However, there was no clear method for transitioning to ALM. Various approaches were developed, primarily by vendors, to support their different methods of selecting the most suitable individual tools that could work together through common interfaces. Eventually, a fully integrated tool was created to cover all the different tasks, processes, and areas of ALM. The providers from the 2000s could manage these technical issues quite readily, but the academic community lacked theoretical and methodological support.

During the years several journal articles were published, however, only about half of the entries were in ranked Journals as seen in Table 2.5. This could be due to a number of factors, such as the fact that ALM is a relatively new field of research, or that it is a field that is not as well-funded as other fields. Though there are numerous entries in this category, means authors addressing this topic for discussions of technical, theoretical, methodological, and business aspects also already in the scientific community. In the top ranking journals (Q1&Q2) there are more publications compared to the bottom half (Q3&Q4), which indicates that high-quality level publications are present and interest is there from the research community. A similar phenomenon is observable for the Conference materials in the Table 2.6. The ALM entry is relatively young and has been dynamically forming in the last decade, thus the lack of clarity and strong base could also result in more publication in ranked journals, as peer-reviewed scientific articles are representing already a high-quality quality committed approach from the researchers with deep investigations and preparations included. Moreover, there is less competition among the high-level journals for this relatively new area, thus more publications are accepted by non-ranked journals. Furthermore, the ALM area is multi-disciplinary, and the boundaries with management, IT, and scheduling might not properly fit for ranked journal scope. Due to such scope alignment, the peer-review process could take up longer time to find reviewers and get the papers accepted. The number of articles published in top-ranked journals has increased over the past few years, suggesting that the ALM research community is becoming more prominent. The distribution of articles across the different journal and conference ranks suggests that there is a need for more high-quality journals and conferences in the ALM field. The ALM definition research can contribute to this target by creating a common understanding of a definition that is usable for further research.

The early appearance of the conference papers indicates and confirms the ALM penetration into academic areas as seen in Figure 2.15. First sporadically started to be discussed and the early results were presented to the research community. Some years later the significant increase in the amount of conference proceedings indicates also that the research community was getting more interest in the ALM field, as also scientific journal articles appeared to be published. After the increase it was steadily present during the years, providing a venue for the community for discussions and collaboration potentials. However, it is usual that the authors present their findings at the conferences first, and only later do they summarize their work in depth in the articles, book chapters, or other forms of publication visible in Table 2.7 and Figure 2.14. Sometimes even years later thus an incorporated shift in publication forms can be accepted as normal behaviour.

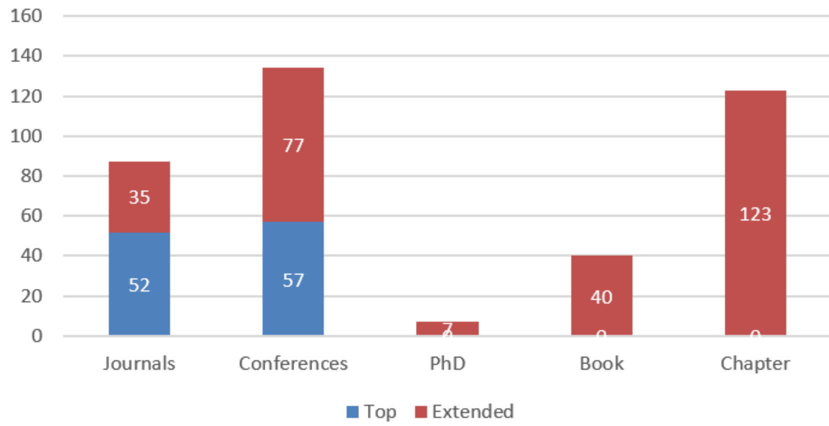


FIGURE 4.2: Top and Extended Academic sources content ratio

The material availability ratio of the ALM discussions in the top and extended academic materials seen in Figure 4.2 also indicates the significant presence in the top tier sources. However, the overwhelming part is in the extended academic ranking. After all, the increasing number of publications shows that the in-depth ALM research was taken by the scientific community, next to the field experts and practitioners, who were also summarizing their work in several books and book chapters over the years.

Over the years, researchers and practitioners have attempted to define ALM, resulting in notable variations in how it is conceptualized and articulated. Explicit definitions availability in the sources is visible in Table 4.1, indicating that only a limited number of sources provide in their works the exact expression for it. The reason for this might be that ALM is inherently interdisciplinary, drawing from fields such as software engineering, project management, quality assurance, and information technology. This interdisciplinary nature results in diverse perspectives on ALM. Authors with backgrounds in software engineering may emphasize the technical aspects of ALM, focusing on tools and methodologies, while those from project management backgrounds may emphasize the organizational and process-oriented aspects. ALM practices continually evolve alongside advances in technology and changes in software development methodologies. Authors who have witnessed these shifts may have different perspectives on what constitutes ALM due to their exposure to various technology stacks and methodologies. These differing viewpoints lead to variations in how ALM is defined and conceptualized as visible in Table 4.2. Interesting to see also the top and extended academic area understanding and usage of the definition of ALM, which is visible in Figure 4.1. It can be observed that the primary conceptualization of Application Lifecycle Management definition in top academic content closely aligns with theoretical paradigms and holistic perspectives, emphasizing a comprehensive approach or methodology. Conversely, the extended academic perspective tends to emphasize the technical dimensions, positing that ALM primarily functions as a tool for managing artifacts, with its procedural aspect representing a comparatively smaller proportion of its overall characterization.

While the divergence in ALM definitions and content can be observed to be present in the ALM community, it is also important to find the common values along that a definition for future research can be provided, see this effort in the following section.

4.1.2 Summary and Conclusion of SLR

The current study explores the Application Lifecycle Management related academic works in Information Systems resulting in valuable contributions being made to the existing literature and future directions for methodological research. Within this investigation, the scientific definitions and descriptions pertaining to Application Lifecycle Management were meticulously examined, drawing upon the extant academic sources in the field. To carry out this research, the PRISMA guideline was adhered, which proved to be an invaluable tool for comprehensively scoping the breadth of available research in the realm of Information Systems. Finally, a critical review proceeded to extract and synthesize the definition of ALM. The discussion section scrutinizes the obtained conclusions in depth.

A significant number of articles failed to provide explicit details regarding the approach and procedures employed, which is considered an unfavorable practice within the realm of scientific research. In addition, it is suggested that future studies should concentrate on determining the order of importance for knowledge synthesis subjects and further refine the principles that can effectively direct the creation and composition of various types of reviews within this discipline.

4.1.3 Synthesizing ALM definitions

ALM is a broad concept that encompasses various aspects of software development and management. The objective of analyzing the content and scope of the ALM definitions and determining the commonalities to be considered to serve as a base for scientific methodical research and mathematical modeling is intended to be an add-on value contribution to the ALM field. Identify the key components or phases of ALM that have been consistently mentioned in the literature, including such as requirements management, design, development, testing, deployment, maintenance, and retirement, keeping the scope and definition evident and tangible enough to enable it to be used in the upcoming researchers, opening new horizons in ALM field.

The life-cycle definition contains two main phases for ALM, a well-defined development phase and a less definable operation or maintenance phase from a scheduling point of view. This means that in the development phase, from ideation to employment, clearly defined SW development can occur; however, after deployment, it can still appear for development tasks and activities.

Scope of ALM ALM is a holistic approach to managing software applications throughout their entire lifecycle, from inception to retirement. It is realized by integrating and managing various activities and work products related to 3 functions such as governance, development, and operations, including maintenance. Governance is an overarching management activity during the whole lifetime of the ALM, however, its importance is higher in the upstream due to its influence factor. Development is mainly related to the classical SW development projects containing the main R&D related work. Operations and maintenance are somewhat similar to service; however, due to the fact that in this phase after the bugfixing, additional non-planned development tasks can appear in different sizes makes it unique.

Phases of ALM There are primary three main milestones for ALM: Ideation, deployment, and end-of-life. There are 7 phases: requirements gathering, design, development, testing, deployment, maintenance, and decommissioning.

Key Components The core components of ALM are for supporting the lifecycle with processes and tools such as version control, issue tracking, continuous integration, and deployment automation. These components play a crucial role in scheduling and resource allocation.

Scheduling Challenges Specific challenges associated with scheduling in ALM exist. These include resource allocation, as activity realization is mostly bonded to finite resources. Task sequencing, as the scheduling of activities in the development and maintenance phase, might need to be handled differently. Time estimation for resources based on scheduling methodologies might be difficult and not obvious. Optimizing resource utilization and scheduling, as currently existing methodologies have not proven optimal for ALM specificities.

ALM Development methodologies ALM is tightly integrated with the software development process. Scheduling within ALM should consider flexible SW development methodologies like Agile partially or fully applied as a Hybrid approach. However, until now, no specific ALM-related methodology or framework has proven to work optimally.

Flexibility and Adaptability ALM scheduling methodologies should be flexible and adaptable to accommodate changing requirements, unexpected issues, and evolving project priorities. Handling and managing changes not only during development but also in the operation maintenance phase. These additional change requests can extend from task level to even smaller subproject levels.

Measurement and Metrics Measuring and tracking key performance indicators (KPIs) related to ALM scheduling is also crucial due to its flexible handling of structure and the necessary contracted values to be contacted. This can include metrics like project duration, resource utilization, and task completion rates.

So as a summary, the ALM definition unified contains the extended definition of time, cost, resources, and quality. The time perspective is covered by the scope and phases, stating ALM is from inception to retirement of the software application, which is divided into 7 phases: requirements gathering, design, development, testing, deployment, maintenance, and decommissioning. The resources are the renewable and non-renewable resources necessary to implement the activities in the scope defined. which allocation is handeled by the key components along the ALM development methodologies, in a flexible scheduling. The quality parameters are defined by the Measurement and Metrics for observing the readiness and maturity levels.

4.1.4 ALM Definition Research Summary

The research questions in this dissertation were focused on available ALM definitions and their characteristics and how a common definition can provide a strong base for future research. For **SLRQ1 & SLRQ2** the extracted definitions and their summary are revealed and shown in Table 4.2 reflecting a total of 7 different understandings. The definitions cover the current academic understanding of the scopes and contents of Application Lifecycle Management.

For **SLRQ3** it was shown the proposed synthesized definition detailed in specific areas for a better understanding of the scope. The conclusion is that ALM has the capability to integrate, coordinate and manage the different phases of the software delivery process, from development via deployment, operations and maintenance. ALM also involves a set of pre-defined processes and tools that include definition, design, development, testing, deployment, and management in a flexible framework capable of handling unplanned and unexpected changes. So in summary this means the results of the ALM definition mentioned above are covering from the Matrix method the logic domain, i.e. the tasks and their relation between during the life-cycle, and can be represented with the cost, quality and resource descriptions. This is resulting then that the Matrix based representation for the ALM is adequate for describing the ALM problem scope.

4.2 Descriptive statistics for Simulation

This section contains the results from the methodological research related to simulation.

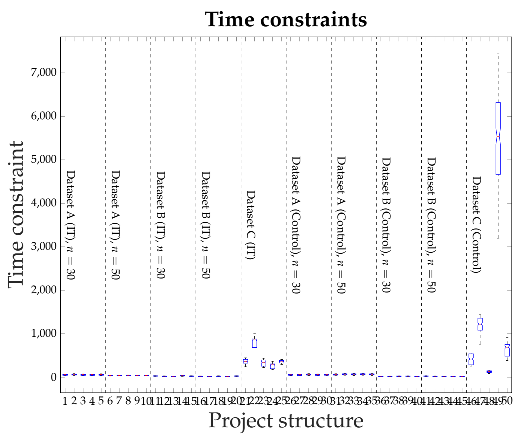
After the descriptive statistics in this section, the answers are provided such as Which project planning and scheduling approaches allow the most projects to survive the changes in task demands and customer requirements. For the survived projects, which project planning and scheduling approaches mitigate most of the effects of project risk and what is the importance of the risk factors to the sensitivity of project schedules. In the last subsection, the threats to validity are discussed.

Figure 4.4 shows the results of the descriptive statistics of 48,000 scheduling problems, which are based on a set of 50 project structures. The project structures of 1-25 consisted of generated and real IT projects, and the control groups (26-50) followed construction project structures. Since 0-50% of task completions and dependencies between tasks are considered flexible, the constraints were calculated individually for each scheduling problem (see Section 3.5.2) Figure 4.4 shows the time, quality, score and resource constraints by project structures and by flexibility parameters. Constraints are specified at $\frac{1}{3}$ and $\frac{2}{3}$ of the theoretical range of project demands. These constraints were the same for all PMas; therefore, they can be compared. However, the specification of constraints fits the possibilities of the project plans. Therefore, it can be confirmed that the actual projects from 3.5.1 have more time and cost demands (see project structures 21-25 and 46-50 in Figure 4.4(a,c)). In that case, the quality demands are also higher (see project structures 21-25 and 46-50 in Figure 4.4(e)). On the other hand, the generated projects (from 3.5.1) have the highest resource demands (see project structures 15-20, 35-20 in 4.4(g)). From the MANOVA cluster, only one project structure (49) is shown to exhibit a relevant difference in constraints (compare 4.4(a) and 4.4(c) and 4.4(k)).

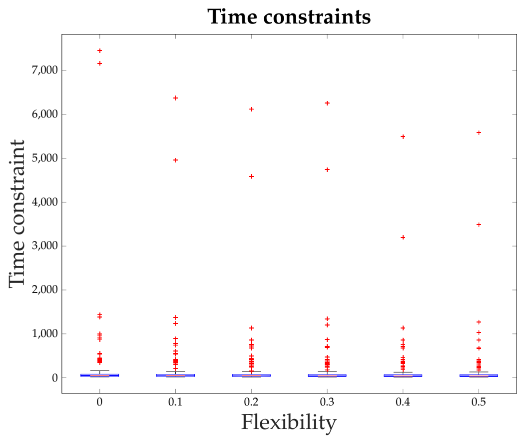
To compare the results of all the risk factors, such as changing the constraints and parameters, as well as the risk effects, such as feasibility and scheduling, the performance is considered in terms of relative values (ratios) (see Eqs. (3.3)-(3.5)). The results show that a risk analysis should also include real-life projects because their constraints may be different from project structures in a standard database. Nevertheless, from the view of project constraints, the difference is lower between IT and non-IT projects. The other interesting results are that if a constraint is calculated by the proportion of a project demand, the absolute values of constraints are relaxed. The results presume that the more flexible projects can be managed with less project demand (see Figure 4.4(d,f,h,j)). However, this can only be true if also flexible methods are used for scheduling.

4.3 Sensitivity analysis

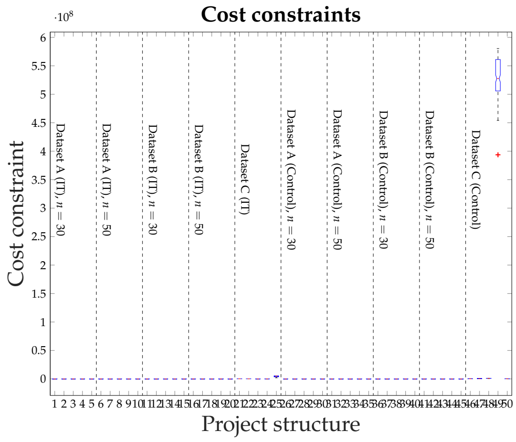
Sensitivity analysis is a technique used to assess how changes in the input variables of a system or model affect its output or outcome. It is a tool that helps to understand the degree to which different factors can impact the results of a decision or analysis.



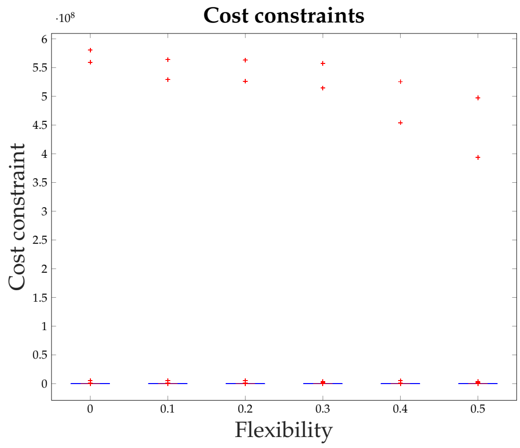
(a) $p = 0$



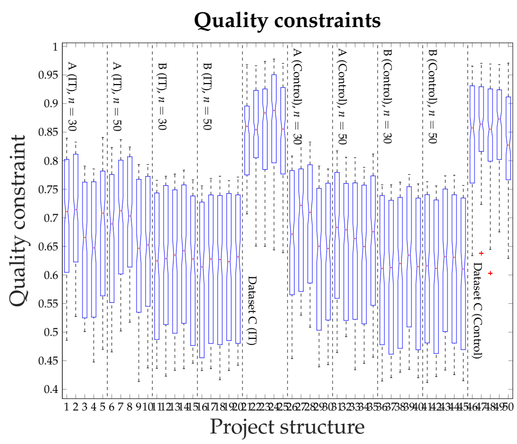
(b) $p = 0.0070$



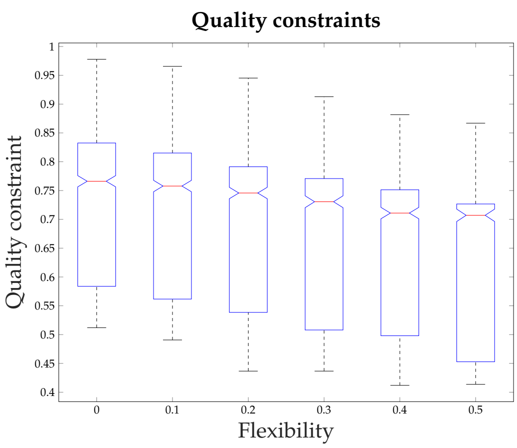
(c) $p = 0$



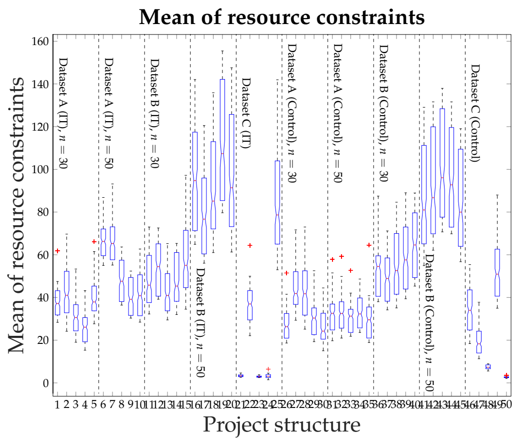
(d) $p = 0.9358$



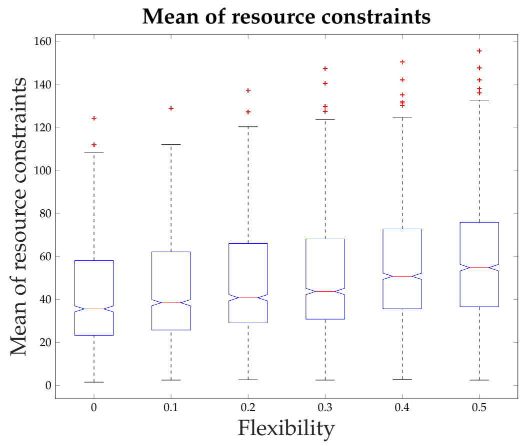
(e) $p = 0$



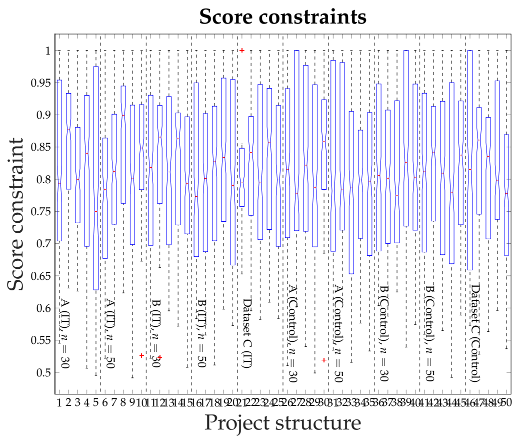
(f) $p = 2.8632e - 159$



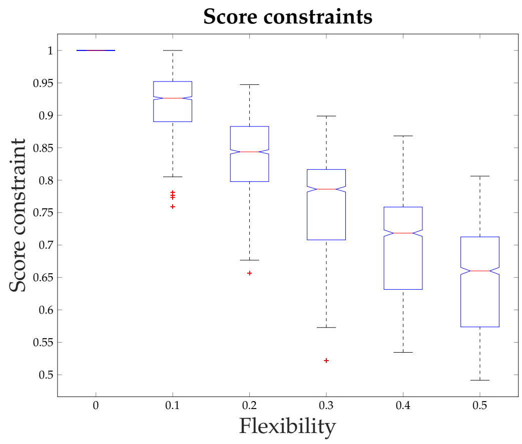
(g) $p = 0$



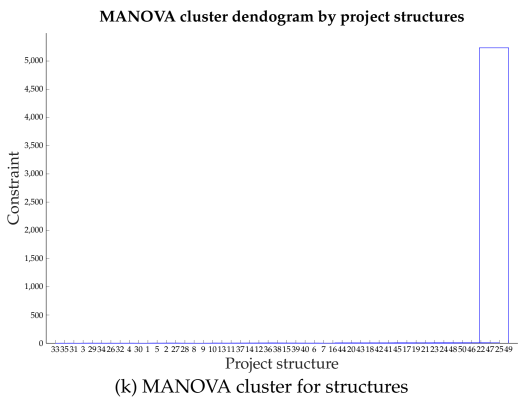
(h) $p = 4.8488e - 67$



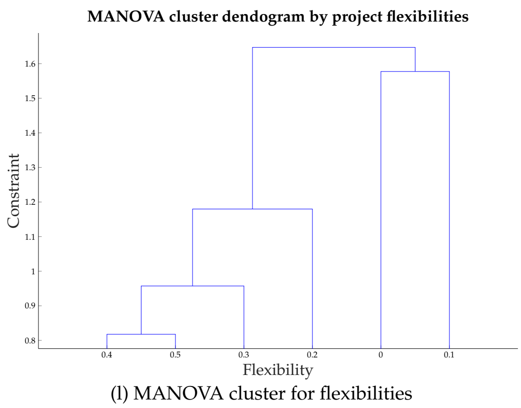
(i) $p = 1.32e - 13$



(j) $p = 0$

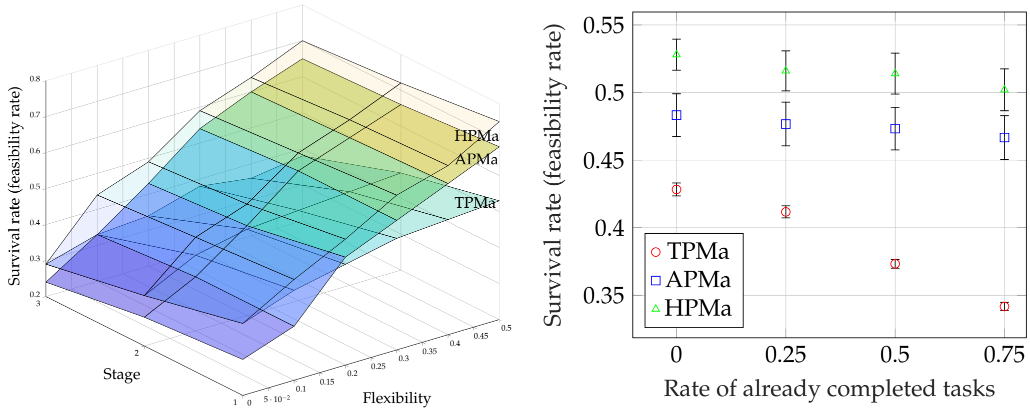


(k) MANOVA cluster for structures



(l) MANOVA cluster for flexibilities

FIGURE 4.4: Results of (M)ANOVA for constraints, project structures and flexibility



(a) Flexibility vs. feasibility

(b) Survivals in the implementation phase (Stage 3)

FIGURE 4.5: Feasibility rate of project management agents by flexibility

It helps decision-makers identify the most important variables in a system, assess the risks associated with different scenarios, and evaluate the robustness of their models or decisions. In practice, sensitivity analysis involves varying one or more input variables of a system or model and observing how these changes affect the output or outcome. In the simulation the interest point was for feasibility as the primary target, then the flexibility, and also to see how the scheduling performance.

4.4 Feasibility versus flexibility

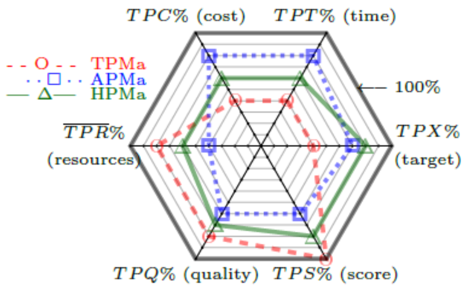
Figure 4.5(a) shows the feasibility rates (i.e., survival rate) of project management agents by stages and flexibility. The survival rate gives the ratio of feasible project scheduling problems in the given stage managed by TPMa, APMa, or HPMa. Stage by stage, increasingly fewer projects survive the changes in constraints (Stage 1), the changes in demands and structures in the planning phase (Stage 2), and in the tracking phase (Stage 3). Especially in Stage 3 (see Figure 4.5(b)), the TPMa is more sensitive to the changes in demands, while the flexible approaches are generally less sensitive (see Figure 4.5(b)), even if the flexibility ratio is high (see Figure 4.5(a)).

In line with Figure 4.4(d,f,h,j), Figure 4.5(a) shows that generally, the increase in flexibility increases the rate of feasibility for all approaches. However, this opportunity can be exploited primarily by agile and hybrid approaches. In addition, in cases of lower flexibility ($< 20\%$), the TPMa manages more feasible projects than does APMa (see Figure 4.5(a)).

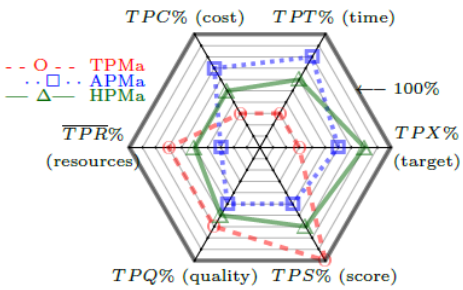
The interesting result is that HPMa made better use of the opportunities offered by flexibility. HPMa makes more feasible projects than the agile approach.

4.5 Scheduling performance

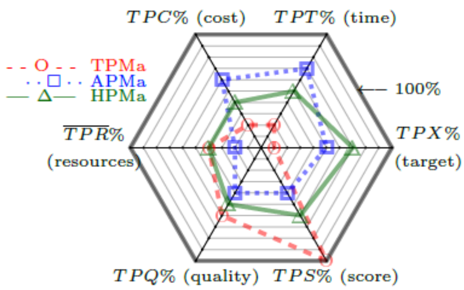
When analyzing the scheduling performance of project management approaches, only the feasible project plans are surveyed. Figure 4.5 shows that HPMa produced the most feasible projects. The agile approach is the second best in the case of a flexible project environment and the third best if there are a few possibilities to reorganize the project or postpone tasks. A similar figure can be drawn for the target



(a) Stage of contracts (Stage 1)



(b) Stage of scheduling (Stage 2)



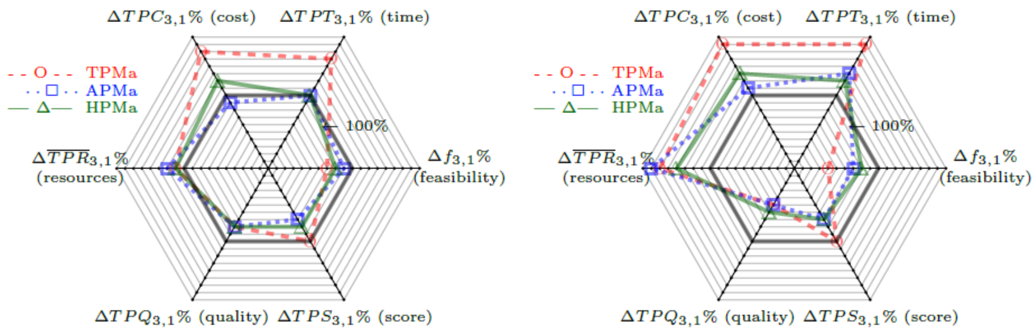
(c) Stage of tracking (Stage 3)

FIGURE 4.6: Scheduling performance of targeted (TPX) and nontargeted (remaining) parameters

functions, but if the remaining parameters are also considered that are not involved in the target function, a much more nuanced picture can be obtained. In Figure 4.6, the $TPX\%$ represents the scheduling performance for the target function. Moreover, $TPT\%$ shows the scheduling performance when the target function was not to reduce project durations. Similarly, $TPC\%$ shows the scheduling performance for cost when the target function was not to reduce costs. According to Eq. (3.3), higher values produce better performance, such as lower $TPT/TPC/TPR$, but higher TPQ and higher TPS. Figure 4.6 shows that HPMa produces the best performance for targets ($TPX\%$) in all stages (83%, 81% and 75%), which means this approach secures the closest to the best total project value. However, the price of this approach is that other parameters are closer to the constraints. Furthermore, TPMa insists on scope; therefore, $TPS\%$ is always equal to 100%. However, the price of this requirement is that TPMa produces the longest projects, from which the risk effect endangers the customer's and management's objectives, and the highest project budget is viewed as unfavorable to management, while the worst scheduling performance is achieved for targets in all stages (41%, 34%, and 14%). Nevertheless, TPMa demands fewer resources per time unit, while parallelization of tasks in APMa and HPMa demands more resources per time unit; therefore, the restriction of the maximal amount of work-in-progress tasks is justified. Therefore, the price of utilizing flexibility is a more problematic resource management issue in agile/hybrid than in traditional approaches. This issue may be increased in a multi-project environment, where parallel projects should share resources with each other. APMa, while capable of maintaining the second place of the scheduling performance in all stages (71%, 64%, and 59%), usually achieved this performance with the shortest projects. It reached the lowest budget if other target functions were selected.

4.6 Performance of risk mitigation

The performance of risk mitigation of the explored project management approaches are swohn in Figure 4.7.



(a) Uncorrelated risk effects (b) Correlated case

FIGURE 4.7: Performance of risk mitigation of project management approaches

The ideal risk mitigation strategy maintains all project plans as feasible, while other $\Delta TPX_{3,1}\% = \Delta TPX_{2,1}\% \cdot \Delta TPX_{3,2}\%$ values stay close to 1.

The TPMa keeps all tasks, and therefore, $\Delta TPS_{i,j} = 1$ in all cases and for all $i > j$, but the price of this strategy is to "lose" more project plans than other strategies. Moreover, considering only feasible project plans, TPMa shows the greatest tendency to delays and overbudget situations. If risk factors are moderately correlated ($\rho \geq 0.6$), the TPMa demands a substantial amount of additional resources. The APMa shows a very different picture. Interestingly, the agile technique is the only approach that reduces project costs despite the risk factors. The price of this strategy, however, is that it attains the largest decrease in quality and scope. It is also interesting that when risk factors are moderately correlated, because of the forced parallelization, the demand for resources is increased to the greatest extent in this strategy. HPMa keeps most project plans feasible, and this approach creates a balance between the multimode methods and the restructuring techniques. Moreover, $\Delta TPX_{i,j}$ is usually very close to one, which means that this strategy can well mitigate the risk effects in order to keep the project plans within the constraints. In the meantime, it retains more of the scope than agile techniques.

When risk factors are correlated with each other, they greatly enhance each other's risk effects. These effects of interdependencies between risk factors occur particularly in the case of using TPMa. TPMa is very sensitive to the changes in the time, cost and resource demands and their interdependencies, which is in line with the experience gained so far in software projects. The agile techniques can better mitigate the risk effects; however, if risk factors are correlated with each other, because of the forced parallelization, this technique is also sensitive to the resources. Furthermore, agile, traditional and hybrid techniques may be useful to different stakeholders (see Table6.1).

4.7 Importance of risk factors

The survival random forest algorithm is used to calculate variable importance (see Figure 4.8). The projects that remained feasible at the end of the simulation stages were those that were considered as survived projects. Moreover, instead of time, the stages of the simulation and the scheduled rate of tasks are considered. Except for the target function ($p = 0.1017$), all variables are significant. The error rate of the model is only 0.0051.

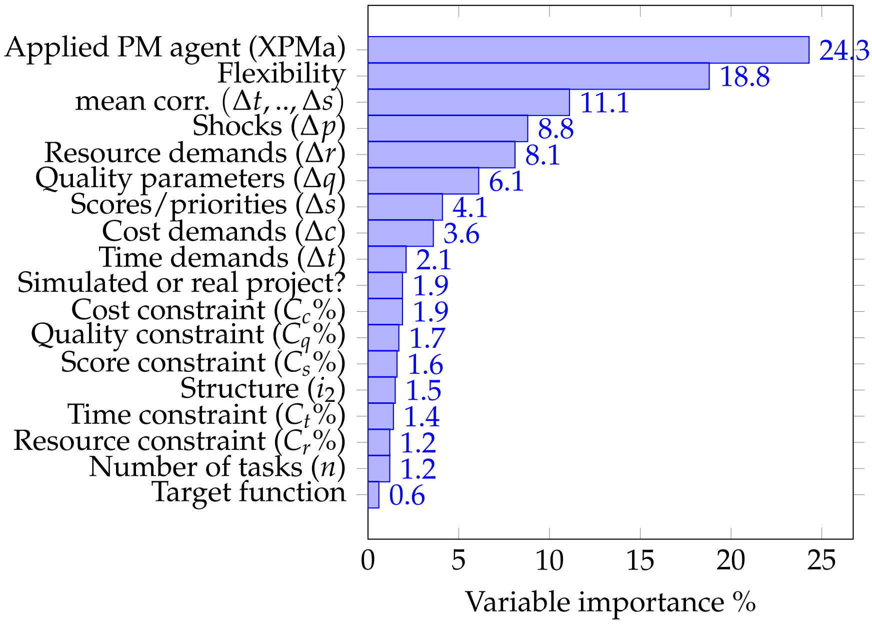


FIGURE 4.8: Variable importance for survivals

Figure 4.8 shows the effect of project management agents; all explored structural properties, such as project structure and flexibility; low-level risk factors, such as changes in costs (Δc), duration (Δt), resource demands (Δr), etc.; and high-level risks, i.e., when TPT, TPC, TPQ, or TPS values violate the corresponding constraint, that are assessed through the constraints (C_x%). According to the result, the low-level root causes and structural parameters have a greater direct impact on survival. The most important variable for maintaining the project feasibility is the selected project management agent (XPMa, 24.3%). In addition, the second most important variable is the flexibility rate (18.8%), which was detailed in Section 4.4. The correlation between risk factors is more important (11.1%) than the risk factors themselves; therefore, a meta-analysis to consider the interdependencies between risk factors is justified. The effect of correlation between risk factors is detailed in Section 4.5. The selection parameter (shock effect, $\Delta p \in 0.1, 1.0$) is the fourth important variable (8.8%). According to the results, TPMa is the most sensitive to the shocks ($\Delta p = 0.1$), where only a few (i.e., 10%) of task demands are changed, but these changes are (even 10 times) higher.

The risk factors ($\Delta r, .., \Delta t$) are more important than the constraints as the result of an agreement (C_t%, .., C_r%). This observation proves that after the contract phase, there are more challenges for the project manager to ensure that the project plan remains feasible. The more challenging task is the resource allocation, both in the traditional and in the flexible project management approaches (see the details in Section 4.5).

The database contained not only IT but also a construction project; therefore, it is an interesting result that the original project structure, regarding the size (number of tasks, n (1.2%)) and i_2, which shows the parallelization, is less important (1.5%). The importance of the data source (simulation or real project) also has low importance (1.9%). The low importance value of the result raises the possibility that flexible approaches can be successful in different kinds of project structures, and if the technology were to allow these approaches, they could also be successful in non-IT projects. This result is explained by the fact that flexible techniques also allow parallelization when they can reorganize the project structure. Therefore, the main question regarding the use of the flexible project management approach, such as agile and hybrid approaches, concerns whether the project plan is considered flexible. Alternatively, in other words, to use flexible project management approaches, the project plans must be flexible. Whether it is an IT or a non-IT project is of secondary concern.

4.8 Novelty of results

Since its emergence in the 2000s, agile project management has garnered the attention of numerous experts who have sought to compare its efficacy with that of traditional project management. Conventional project techniques are considered the origin of formality in project management and have been utilized for an extended period. The scholars emphasize the success of certain industries. However, for complex projects, particularly those related to IT and software, traditional methods may not be as effective. This is because the requirements for such projects are intangible and subject to change, making the iterative and customer communication-focused agile approaches more suitable and successful.(Salameh, 2014; Gaborov et al., 2021)

Then the combination of traditional and agile project management, a hybrid approach was also in the focus, as visible the agile changeover was only partially possible for organizations, or they wanted to react to the fast-changing requirements with agile practices introduction.(Grey, 2011; Adelakun et al., 2017; Gemino et al., 2021) The efficiency for such changeovers are also examined by several scholars looking for the organizations and management styles to be aligned, where it is senseful of the combination resulting the hybrid approach (Papadakis and Tsironis, 2020; Diem et al., 2021; Leong et al., 2023).

However, the exact performance preparation of the methodologies based on simulation or real-life data is scarcely available, this is why the results from the matrix-based simulations are important.

In terms of scheduling, the traditional project management approach and the implemented TPMa operate only in terms of multimodes of task completion. This approach assumes that tasks can be completed in different kinds of ways. In contrast, agile techniques assume a flexible project structure, where dependencies between tasks can be flexible and lower-priority tasks can be postponed until the next project, but usually, only one completion mode is specified. The results showed that in the case of a flexible project environment, where the flexibility rate is high, this approach can truly produce more feasibility, and in this way, it can make remarkably more projects capable of success than traditional approaches. However, this advantage dissipates when the technology requires strict dependencies.

Hybrid techniques allow both multimodes and flexible structure and therefore, it is assumed that this is the supreme technique of project management. This assumption is reinforced by the fact that this technique provides the highest ratio of feasible solutions and the best scheduling performance when only the target function (see Table 6.1) was considered. To answer List of Tables, based on the proposed database, HPMa provides the most feasible solutions; therefore, ***a software development project is more likely to survive the risk effects if a project plan is managed by a hybrid project management approach***.

Currently, the flexible project scheduling algorithms are much less sophisticated than the trade-off methods or the MRCPSP algorithms. For example, there is currently no multipurpose version of agile or hybrid scheduling, and only one target is considered in scheduling and risk mitigation. Table 6.1 shows the ranks in addition to the scheduling and risk mitigation values. The results show that the HPMa does not usually mitigate the risk effects the best (see List of Tables). Nevertheless, selecting an adequate project management approach and ensuring the project flexibility (see Figure 4.8) are the main factors for both feasibility, and performance of scheduling and mitigation.

Notwithstanding these findings, because of technical requirements, there are substantially more obligatory dependencies between tasks, and the flexible project management approaches do not achieve better performance.

Nevertheless, to answer List of Tables, the most important variable for project survival is to select an adequate project management agent, but the second most important variable is ensuring flexibility. The flexibility parameter is much more important than the other structural parameters, such as the project size or the number of work-in-progress (WIP) activities, which are very limited in flexibility, especially in agile project management approaches.

As a summary for Chapter Results, it was shown that the systematic literature review was a significant contributor to the thesis itself with the definition extracted, and after the critical review part used for answering the RQ1. The contribution with this process also helped the academic literature to have the first systematic literature review for the ALM definition.

For the methodological research part results were also presented, namely the feasibility and performance of the project management scheduling approaches in the ALM environment. The sensitivity analysis of the results are showing that the selection of the proper method can influence significantly the performance.

Finally, also the scheduling related risks were shown and analysed for based on the simulation results.

Chapter 5

Validation

This chapter demonstrates validation for the simulation results with a Case-study from an ALM environment in the automotive industry. The Case-study is demonstrating and Application development environment, which is using Agile methodology for work.

5.1 Case-study description

Resource and organization point of view, within the R&D organization the customers are handled by integrated teams. This means from competencies within the organization there are dedicated team members for the customers. Based on their resource demand it is possible though that they are working not only for one but more customers also due to the matrix organization, as it is cost- and resource-usage efficient. Usually, there are 12-15 core team members who work closely together, led by a customer project manager focusing on the customer projects and acting as competency project managers also, which means in the background they are keeping the connection with the field experts who are involved on-demand only.

Process and Quality control point of view, the company is dealing with safety-critical product development in an automotive area, which requires several standards to adhere to also, e.g., ASPICE (Messnarz et al., 2018), ISO 26262 (Y. Fang et al., 2023), and IATF 16949 (Yadav and Heriyati, 2023). The internal processes are compatible and appropriate to the international standards, thus the team is encouraged to keep them primarily during the V-model-based development. There are Technical releases and Product releases defined to prove that the required maturity levels are fulfilled and documented to the defined customer milestone gates. Technical releases are carried out on the competency level (SW, HW, Safety, System) and Product releases happening on the Project level.

Customer-specific projects approach is present also, which means from traditional project management there are significant differences which are moving the application software project management towards application lifecycle management. Due to customer requests partial agile activities already introduced in SW development such as biweekly SW delivery, support for Agile work products, Product Increment (PI) planning participation, Post SOP (Start of Production) SW delivery for milestones, additional variants support after the first production milestone, new features and functionalities later introductions. Some of the OEMs adopted SAFe organization already, thus it is requested to support their milestones and procedures also as extra tasks from the R&D organization. For example, a platform at OEM requires application software in the time horizon from 2019-2045 visible in Figure 5.1. The Contracting and alignment period started in 2019, the main development with continuous development requirement evolution and implementation from 2020, first production release (SOP1) in 2021. Post-SOP development still happening: bugfix,

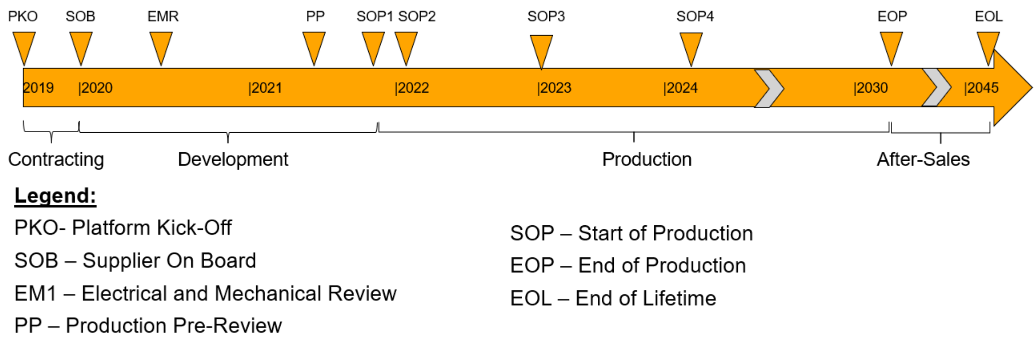


FIGURE 5.1: Main Vehicle OEM Milestones Overview - Own edit

extra feature request, an adaptation of existing feature, new carline introduction on market demand, legal regulation update, and several additional, unexpected tasks on the platform. End of production is expected in 2030, however, in case there is a breach for the Cyber Security part of the SW for example, then an update and rollout for SW modification potentially will be necessary till 2045, which is the ultimate end of lifetime, retiring of the platform.

So it is visible, that the classical project understanding is not properly fitting anymore for such situations, thus an extension is necessary for the proper and efficient handling of such application lifecycle management.

5.2 Data collection

Data was gathered from primary and secondary sources to ensure the quality of information. For primary data sources, there were experts from various positions and levels (Project Managers, Competency Managers for SW, HW, Safety, SW developers and Testers) and aligned with them multiple times during the preparations and case study execution. As secondary sources, the project databases and descriptive project documents were used, such as project plans, schedules, issue ticketing system, release work products, intranet, and version control system to gather and analyze the data so that the model can be created.

ALM plan attributes	min	typical	max	unit	avg duration [typical version]	
Average duration of ALM (platf. variants)	14	24	36	[month]	24.0	[month]
Average themes within the period	1	2	3	[theme]	12.0	[month]
Average epics within the period	1	4	6	[epic]	6.5	[month]
Average features within the period	3	12	18	[features]	2.2	[month]
Average sprints within the period	9	36	54	[sprints]	21.8	[day]
Average stories within the period	90	360	540	[tasks(stories)]	2.2	[day]

TABLE 5.1: Main construction elements in the ALM - Own edit

As already mentioned the company is using agile-related methodologies, also for the project plans and schedules it is visible, that the partitioning is following the Agile work breakdown, see in Table 5.1. The average duration of the ALM flow can be identified as the customer platform variant lifetime. From high-level to lower-level construction Agile planning has its Themes, which function as an umbrella, and contain strategic initiatives. They describe the high-level direction for the development work that will help you realize your goals. A theme is the largest unit of work in agile development. Below themes are the epics, which are a large body of work, major areas of work across multiple competencies. Smaller units are the features, which are the functionalities, defined by the customers. And to precise these features there are the user stories, which are actually use-case descriptions that are realized in the sprints that last 2 weeks of work packages. The user stories are broken down into tasks, which are executable units for the team already. In the case study, a programmer developer and a tester resources were assumed which are the two main roles identified in the team.

Traditional project-level plans contain all the features, assigned to the defined milestones and they are already defined in advance before the starts. Nevertheless, each project comprises various activities that may not necessarily need to be implemented, and certain features can be subject to change in response to market demands or other constraints. Within the project's sprints, supplementary tasks with flexible dependencies can be rearranged or delayed, depending on the completion of features. An example of this flexibility is evident in new customer-defined features or testing activities. For the new features, only high-level information is available and implementation is also might be basic at first. Also, some tests can be postponed during the early stages without a significant impact on quality or technical debt, leading to a lower priority for thorough testing initially. However, as the project matures, feature implementation and also testing becomes a focal point and becomes mandatory as the final software release approaches. Unlike testing, the likelihood of major architectural changes decreases as the software reaches a higher level of maturity. Additionally, dependencies between activities can be eliminated if permitted. These adjustments in tasks and dependencies can be attributed to internal or customer decisions, allowing management to have greater flexibility. The introduction of these flexible tasks and dependencies has resulted in a new matrix-based flexible project plan from the data available from the company sources.

Timescale	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	Start Day	Finish
Platform #1	——	——	——	——							0	576
Platform #2			——	——	——	——	——	——	——		288	1296
Platform #3					——	——	——	——	——	——	576	1440
Platform #4		——	——	——	——	——					144	864
Platform #5						——	——	——	——	——	720	1440

TABLE 5.2: High-level overview of ALM platforms in case study

So based on the consultation with company experts and management stakeholders, the roadmap for the company environment was modeled at a high level, which can be seen in Table 5.2. The platforms represent the vehicle variants with the closest configuration and the same electrical and physical architectures, the differences are in the vehicle appearance and configurations from combustion, transmission, and brake configuration point of view in these cases. Each platform has its own lifecycles, and the functional and content exchange among the platforms are also highly likely during the lifetime of the platforms.

A more detailed approach for the platform depiction can be found in Appendix B. For the description of the whole matrix, the Application Lifecycle Management Domain Matrix (ADM) is introduced, as shown in Figure 5.2, with a similar structure to the PDM described in the previous chapters. The description contains two lifecycle maturity phases: the first is the development phase, and the second is the operations phase. The Development phase contains the Plan, Develop, Build, and Test sections, which are further divided into Tasks, which are representing key activities. The Operations phase contains the Release, Deploy, Operate, and Monitor sections, which are also divided into further Tasks. The categorization of the tasks are also denoted by colors for easier overview. The orange color represents the *Mandatory tasks*, which means their execution is a must. The dark blue color represents the *Optional tasks* which are supplementary only, means that their execution has an assigned probability, and the Agile and Hybrid agents can decide on their execution. The light blue color denotes *additional tasks*, which might appear as add-on activities during the execution. These *additional tasks* are basically the extension for the simulation representing the ALM characteristics id est appearing unexpected activities in the schedule, which are the most important feature of the ALM approach compared to the traditional project-based definitions. Within the simulation such additional activities are limited on task levels, however, in real extended understanding not only task level but also several related tasks, subprojects might come into as extensions. For the current first simulations and their evaluations, it was decided to stay on task level only, further extensions are for the future planned. ADM contains similarly the TD (time domain) and RD (resource domain) part, which was described in the previous chapters more in detail already. T1-T3 denotes the execution modes, where T1 uses less resources thus a slower execution of about 20%, T2 denotes the normal execution, and T3 uses more resources and thus a faster execution way with 15%. In this case study, two resources are defined, both of them renewable. The R1 is denoting the programmer developer and the R2 is the Tester. The hourly rate for the programmer is higher, in the case study simulation assigned 15 cost units, and the tester is lower, assigned 12 cost units. The cost domain (CD) is calculated then from the resource modes and time modes product.

For the ADM complete setup please refer to the Appendix B where all the 5 platforms are described on task level.

5.3 Simulation environment

For the realization of the ALM environment described above, The Matlab integrated software development tool was used (Mathworks, 2023). An extension of the metaheuristic project solver of Kosztyán (available at `https://github.com/kzst/GENALG_PDM/`) was developed for the ALM-related simulation runs available also online on GitHub repository at `https://github.com/jakabr86/alm-dissertation/`. After the simulation setup, all related entries were defined to cover the use case. For the sensitivity analysis, specific ranges were predefined to have comparable results.

5.4 Result Data analysis

The case study offered valuable insights into the Application Lifecycle Management phenomena within a real-life setting. The results of the analysis were thoroughly discussed with relevant experts and managers within the organization to ensure

ADM (application lifecycle management domain m		TD			RD							
		T1	T2	T3	R1			R2				
Lifecycle	Phase	Time [slower]	Time	Time [faster]	Resource [prog. less]	Resource (progr.)	Resource [progr. more]	Resource [tester less]	Resource [tester]	Resource [tester more]		Task
Platform #1		0	0	0	0	0	0	0	0	0	1	Pre-kickoff activities
DEVELOPMENT	Plan	33	26	21	4	5	6	5	5	5	2	Backlog planning
		28	21	20	4	5	6	4	5	6	3	Analyze impact + review backlog / timeline
		13	10	8	4	5	6	4	5	5	4	Resource and budget estimation
		6	5	4	1	1	1	1	1	1	5	Customer acceptance and resource organization
		6	5	4	2	2	2	1	1	1	6	Design and architecture (carry over)
		6	5	5	2	3	4	2	2	2	7	Kickoff and handover to development
	Develop	63	53	40	4	5	6	3	3	4	8	Develop code + interfaces
		131	106	82	3	4	5	3	3	3	9	Develop feature sets
		52	42	37	3	3	4	2	2	2	10	Develop optional feature sets and parameters
		20	15	13	2	3	3	1	1	1	11	Interface cross-check, generation and compilation
	Build	27	21	16	2	2	3	1	1	1	12	Run config tool, synch models and drivers
		19	15	13	1	2	3	2	3	3	13	Resolve smoke test issues
		28	21	19	2	2	2	2	3	3	14	Target and simulation (debug/dll) build
	Test	47	37	33	3	3	3	4	5	5	15	Setup existing environment
		25	21	17	2	3	4	4	4	5	16	Failure free environment setup
		12	10	8	1	1	1	2	2	2	17	Functional testing: startup test
		54	42	34	2	2	2	3	3	4	18	Automated HIL package #1
		45	35	29	2	3	4	4	4	5	19	Regression manual test
		52	42	38	2	3	3	4	5	6	20	Vehicle test and performance
		12	10	8	1	1	1	2	2	2	21	Functional safety delta and signoff
		42	31	29	1	2	3	2	3	3	22	Validation
OPERATION	Release	17	15	14	2	3	4	2	3	3	23	Package and gate review
		13	10	9	1	2	2	2	2	2	24	OAT (operational acceptance test)
		41	32	25	4	4	4	2	2	2	25	Fix failed items
	Deploy	13	10	8	1	1	1	2	3	3	26	Tooling and package test
		30	24	19	2	2	3	3	3	3	27	Debug and resolve tooling issues
		25	21	16	1	1	1	1	1	1	28	Batch deploy
	Operate	35	26	23	2	2	3	3	3	3	29	Configure and verify backup
		12	10	8	1	1	1	2	2	2	30	Upgrade new version
	Monitor	12	10	9	2	2	2	2	2	2	31	Collect and document findings

FIGURE 5.2: Application Lifecycle Management Domain Map for Platform 1

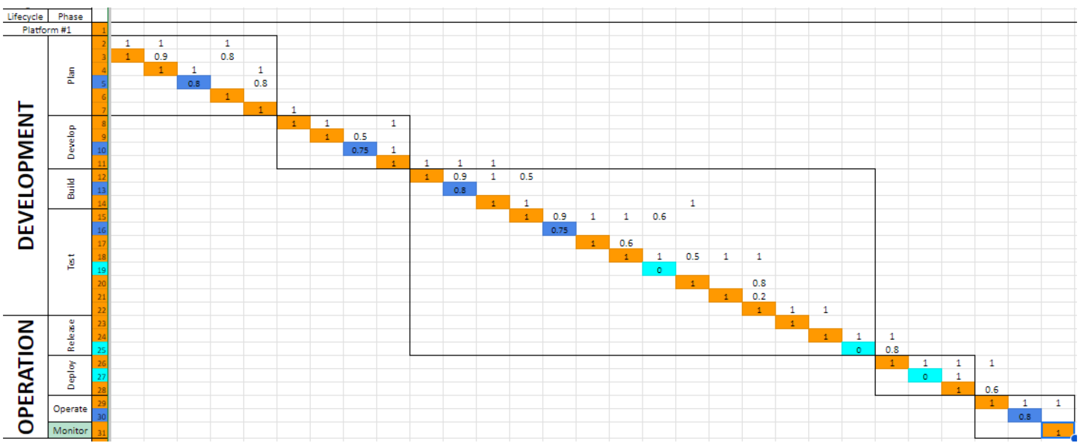


FIGURE 5.3: Application Lifecycle Management Domain Map connection for Platform 1

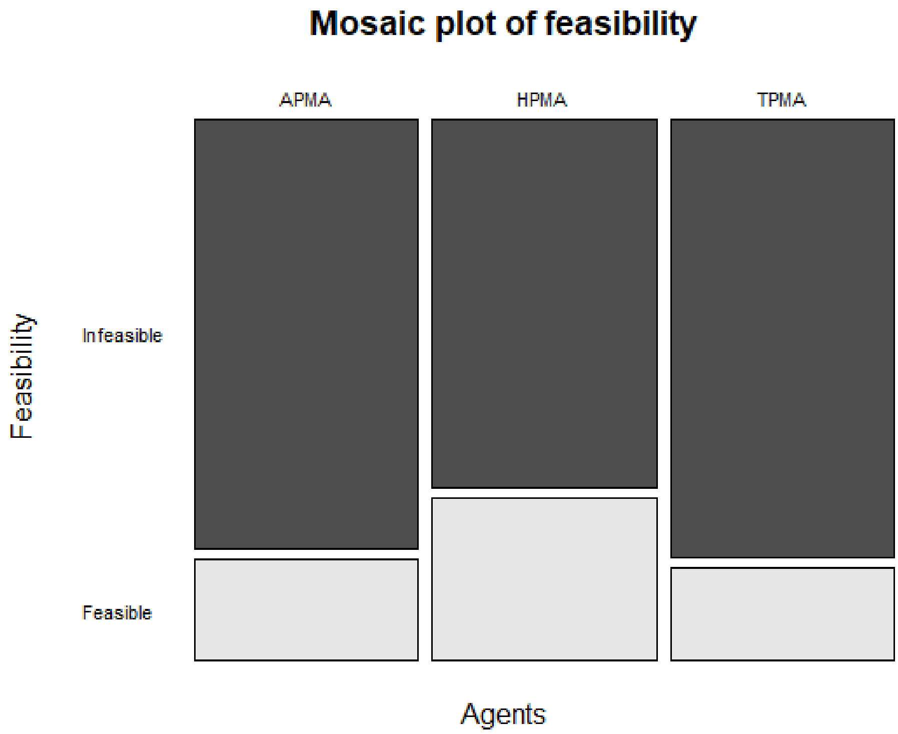


FIGURE 5.4: Simulation results - feasibility and infeasibility ratios for the different agents

their accuracy, gather feedback, and minimize any potential errors or psychological biases. The parameter values obtained from the company's plans confirmed the defined ranges utilized throughout the simulation process and aligned with empirical observations during data collection. One significant discovery from the case study is that the company does not directly consider the relatively high available flexibility ratios, at least not at the planning level. The changes are expected, however, managed only on-demand. With the utilization of the proposed simulation and optimization framework, it becomes possible to effectively harness this flexibility and enhance the company's replanning processes.

Feasibility for Agents					
TPMA		**APMA**		**HPMA**	
Feasible	Infeasible	Feasible	Infeasible	Feasible	Infeasible
110	515	150	475	254	371

TABLE 5.3: Summary of feasible and infeasible results of each agent

The case study simulation results first descriptive results can be seen in Figure 5.4 and Table 5.3.

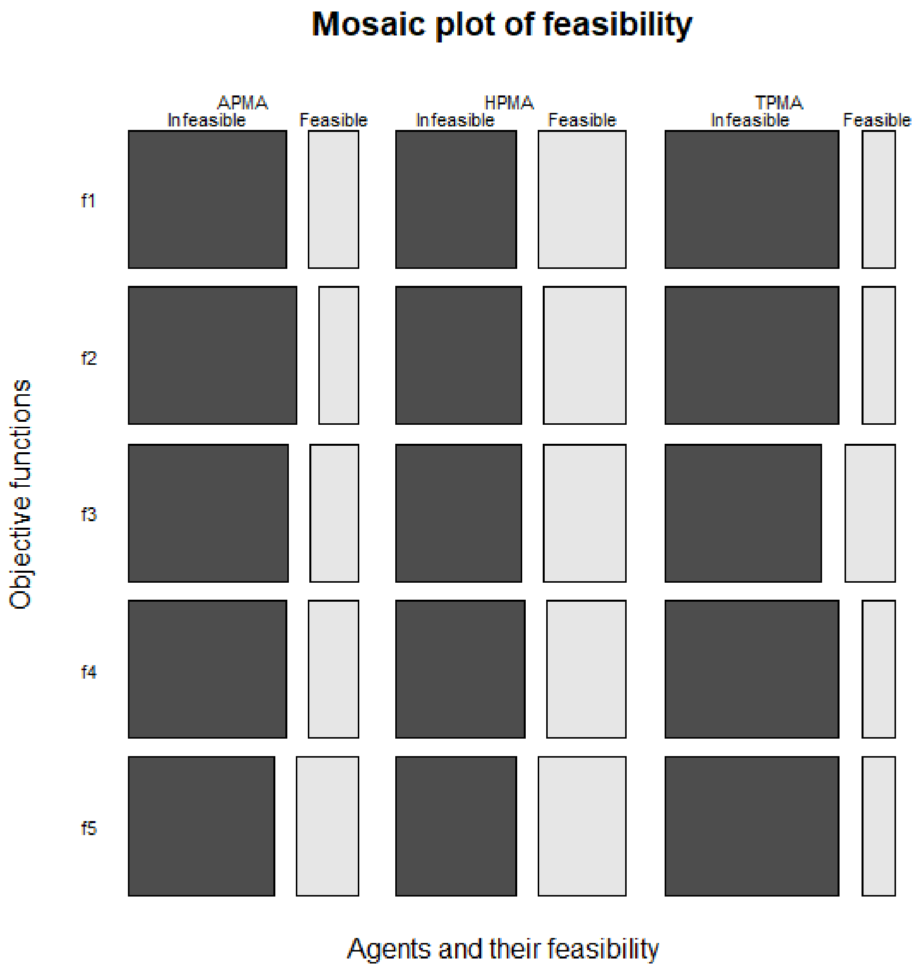


FIGURE 5.5: Simulation results - feasibility and infeasibility ratios by target functions for the agents

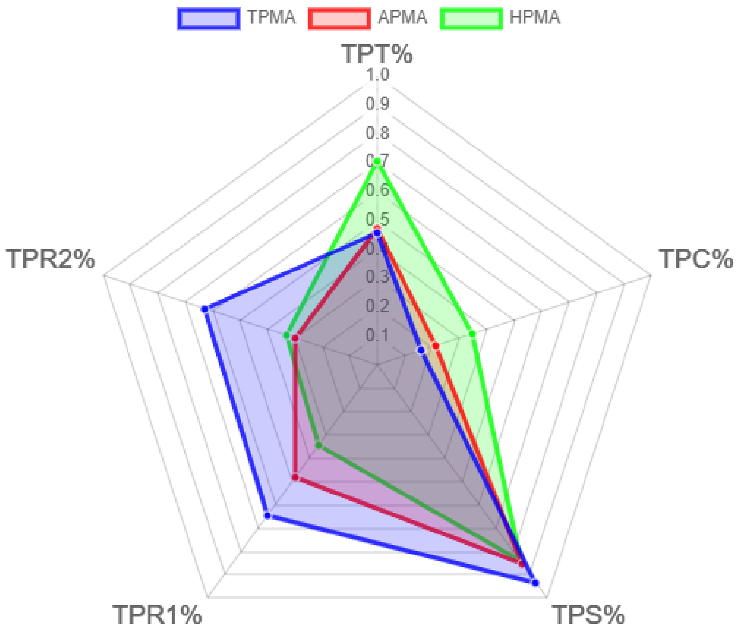


FIGURE 5.6: Radar chart for the performance of the agents for ALM

The case study simulations proceeded with the different target function orientations, where $TPT \rightarrow$ min denotes the target for minimum throughput time, $TPC \rightarrow$ min means the minimal cost, and $TPS \rightarrow$ max is the maximum score target. The distribution of results is shown in Figure 5.5. The results show that the HPMA is overperforming TPMA and APMA approaches in the number of feasible solutions. The second is the APMA and the worst in feasibility is the TPMA. This is not surprising, as TPMA is forced to proceed with all the tasks, only the demands could be changed not the structure. APMA is performing better as it can change the structure and reorganize the tasks. HPMA can change in the demands and in the structure also, see summary in Table 3.2

Agent/Function	**Time**	**Cost**	**Score**	**Resource 1**	**Resource 2**
TPMA	Worst	Worst	Best	Best	Best
APMA	Second	Second	Second	Second	Worst
HPMA	Best	Best	Worst	Worst	Second

TABLE 5.4: Summary table for the performance of agents for ALM

The performance of the agents in the ALM environment is visible in Figure 5.6. Here only the feasible plans are included. In general, for the values on the axes, the higher is the better result. For the Score value, the higher the better, and the remaining axes also contain scaled information:

$$TPX^{\text{scaled}}_{xPMA} = 1 - \frac{\overline{TPX}^{xPMA} - \min(TPX^{xPMA})}{\max(TPX^{xPMA}) - \min(TPX^{xPMA})} \tag{5.1}$$

On the axis TPS% the best result is coming from the traditional (TPMA) approach, as there all the tasks are proceeded, there is no exchange during the process, therefore it is reaching 100% id est $TPS \rightarrow$ max value (0.85496). However, also APMA and HPMA are almost reaching 90% overall scores.

In total cost (axis TPC%) the best performance is achieved by the hybrid approach (HPMA), and then APMA and TPMA are lagging behind, close to each other. Seemingly in this environment, the agile approach cannot perform well in a cost perspective as the resource usage in the restructuring demands higher efforts. HPMA though can harvest its advantage in that it can restructure and also modify the demand parameters.

For total time (TPT%) the best results are from HPMA also with 70%, very much comparable results around 50% value for APMA and TPMA.

For resource utilization, the TPR1% and TPR2% axes are relevant, it is evident that TPMA overperforms the APMA and HPMA. In TPR1% HPMA is the lowest performer with 30%, and APMA with 45%. TPMA performing significantly better, above 60%, which is roughly double the HPMA approach, due to the fact that APMA and HPMA are restructuring modifications that demand higher resource usage compared to the TPMA. For the TPR2% APMA is performing slightly worse with 30%, like HPMA, that also slightly above this value, while TPMA is above 60% here too which is the best value on this acis. Overall, the TPMA approach is the best performer in resource usage, lacking the extra efforts for restructurization and proceeding with the linear approach.

In summary, for the cost and time performance, HPMA is performing the best, while for the resource utilization, TPMA has significantly better results, and a slight advantage for the score also. A summary table can be seen in Table 5.4.

	$\overline{TPT}$	$\overline{TPC}$	$\overline{TPS}$	$\overline{TPR_1}$	$\overline{TPR_2}$	$\overline{TPR}$
APMA	1,445.170	423,594.5	0.7829629	25.38000	31.98667	25.38000
HPMA	1,304.966	403,958.4	0.7740647	28.96850	31.13780	28.96850
TPMA	1,458.369	432,909.7	0.8549600	21.56364	24.64545	21.56364

TABLE 5.5: Descriptive statistics: feasible solutions of agents for each target

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Agent	2	7.7794e+10	3.8897e+10	63.43	0.0000
Residuals	511	3.1338e+11	6.1327e+08		

TABLE 5.6: Result of the analysis of variance

Table 5.6 shows the summary results for ANOVA. The method compares the means of the achieved results of each target (continuous dependent variables) across 3 agents (independent categorical variables). Based on this, there is strong evidence to reject the null hypothesis that the means of the dependent variable are equal across all agents. The very low p-value suggests that the differences in means between at least one agent category and the others are statistically significant.

		n	χ^2	df	p	Method
1	TPT	514	223.34	2	p<2.2e-16	Kruskal-Wallis
2	TPC	514	264.46	2	p<2.2e-16	Kruskal-Wallis
3	TPS	514	123.39	2	p<2.2e-16	Kruskal-Wallis
4	$\overline{TPR}$	514	113.26	2	p<2.2e-16	Kruskal-Wallis

TABLE 5.7: Result of Kruskal-Wallis rank sum test

Based on the very low p-values for all targets (TPX), there are significant differences in the medians of the target functions. There is evidence to reject the null hypothesis that the distributions of the achieved targets are equal across all agents. The agent categories have a statistically significant impact on the targets. To also know which pairs of groups are different, pairwise Wilcoxon tests suitable for non-parametric data are performed with (Bonferroni) corrections for multiple testing.

Target	**Comparisons**	**Significance**
TPT	TPMA vs. HPMA	p<2e-16
	APMA vs. HPMA	p<2e-16
TPS	TPMA vs. APMA	p<2e-16
	TPMA vs. HPMA	p<2e-16
TPC	TPMA vs. HPMA	p<2e-16
	APMA vs. HPMA	p<1.7e-11
$\overline{TPR}$	TPMA vs. APMA	p<3.7e-10
	TPMA vs. HPMA	p<2e-16
	APMA vs. HPMA	p<1.6e-06

TABLE 5.8: Summary of significantly different agents for targets

Table 5.8 shows that TPMA and APMA are distinct from HPMA regarding TPT performance, but TPMA and APMA do not differ significantly. TPMA is unique compared to both APMA and HPMA for TPS, however, APMA and HPMA do not significantly differ. For average resource demands (TPR), all agents perform significantly differently.

Assumption	Outcome
Data type(s)	Satisfied
Sample size	Satisfied
Normality	Not satisfied
Outliers	Not satisfied
Homogeneity of variances	Not satisfied

TABLE 5.9: Summary of verified assumptions

TABLE 5.10: Simulation parameters

Parameter	**Values**
Agents	TPMA, APMA, HPMA
Objective(target)functions	$\overline{TPT} \rightarrow$ min $\overline{TPC} \rightarrow$ min $\overline{TPS} \rightarrow$ max $\overline{UF} \rightarrow$ min Composite
Time constraint (ratio)	100%, 87.5%, 75%, 50%, 25%
Cost constraint (ratio)	100%, 87.5%, 75%, 50%, 25%
Score constraint (ratio)	0%, 12.5%, 25%, 50%, 75%
Renewable resource constraint	RR_1: 38 [unit] RR_2: 40 [unit]

Table 5.10 provides an overview of the parameter combinations used as input for the simulation runs. These parameters cover various aspects of the simulation, including the choice of agents, objective (optimization) functions, and constraints,

given explicitly or as ratios which are calculated from the theoretical maximum. Using these combinations allows the exploration of various realistic scenarios in the parameter space and their effects on the simulation's results. The parameters are selected carefully for adequate sample size and fair competition between the agents without bias, which is also reviewed by experts from the company. The total number of necessary simulations considering all parameter combinations can be calculated as the following: $3(agents)$ x $5(targets)$ x $5(levels)^{3(varied\ constraints)}$ x $1(fixed\ constraint)$ = 1,875 runs. The results were also verified empirically.

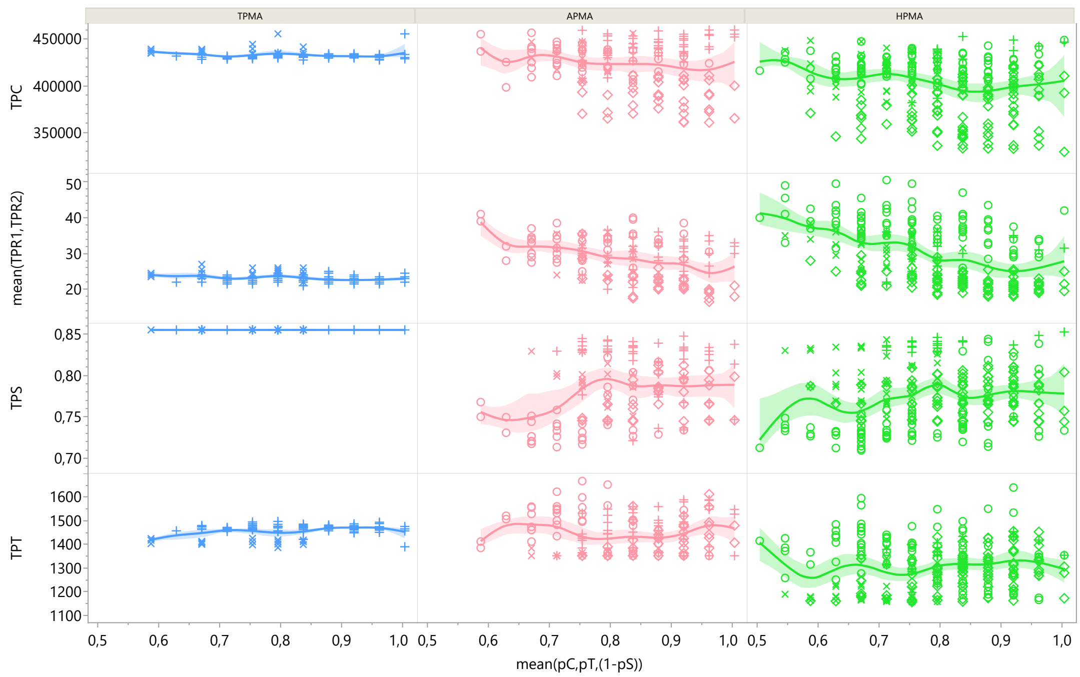


FIGURE 5.7: Insights for agent performances

Figure 5.7 shows each agent's performance in detail. On the horizontal axis, all constraints are represented with their average ratio, where a smaller value means a more strict constraint combination, and similarly, a higher value means an easier constraint set. Only feasible results are considered.

Firstly, TPMA demonstrated a consistent pattern in its results, suggesting reliability in its predictions. However, it tended to be less feasible compared to APMA and especially HPMA, when constraints became stricter.

APMA, on the other hand, showed an interesting dynamic, at least, in terms of duration (time). Initially, it performed less favorably compared to TPMA. However, as the constraints became more strict, APMA improved and successfully increased feasibility, exhibiting results with trade-offs for score, cost, and resources. Additionally, APMA exhibited greater variability in its results. There is a specific range, where APMA is producing more feasible solutions and still gives acceptable, alternative results for the different goals.

Finally, HPMA emerged as the most robust agent. It consistently provided feasible solutions even in challenging scenarios, even where TPMA and APMA were not feasible at all. Variation is also high in the results which makes prediction harder for this agent.

The difference between TPMA and APMA performances suggests that by utilizing multiple modes effectively, the company could have good results even with

TPMA, which means that APMA is not an absolute necessity when constraints are extreme, e.g., either too strict or not strict at all.

5.5 Threats to validity

Threats to validity needed to be carefully considered throughout the research process. Potential threats can be internal, external, construct, and conclusion validity (Brewer and Crano, 2000). For the identified threats, actions were defined to mitigate them which will be discussed in detail in the following sections.

In the first part, the methodology simulation validity threats are examined and determined counteractions as follows:

Internal validity threats can affect the independent variable concerning causality. To avoid such a threat, the following actions were taken:

- *Exploring multiple groups:* IT-projects and non-IT-projects were separated into two groups. Nevertheless, the homogeneity of project structures is explored (see Section 3.5). The selection of project structures was based on former studies (see, e.g., Vanhoucke, 2012). Selection criteria are applied; see Section 3.5.1. However, a new project database should contain quality and score values for testing flexible approaches.
- *Treating missing variables:* Although quality and score values were missing from every dataset, they have been generated according to former studies; see Section 3.5.

The *external validity* involves the possibility of generalization of results outside the scope of experimental settings. To improve external validity, a real-life project database was included. In addition to standard (see Dataset A in Section 3.5) and generated (see Dataset B in Section 3.5) datasets, a real-life dataset Dataset C is considered for the simulation. Further project structures can be investigated if the required parameters exist and/or can be calculated/simulated. Since the dependency and flexibility scores cannot be observed in real examples or obtained from standard databases, the survival ratio of the projects can change, but the effect of the flexibility can be studied due to the wide range of the flexibility ratio.

Construct validity threats may be due to the simplifications made in the software project process modeled for the optimization and the inappropriate application of simulation. To mitigate the effects of such threats, the following actions were taken.

- *Applied exact methods:* Agents are based on exact methods, which guarantee optimal solutions. Therefore, not only the feasibility but also the scheduling performance can be explored.
- *Applied distributions:* Variables (risk factors) in the sensitivity analysis that are based on Monte Carlo simulations followed the β-distribution, which is used in practice (see Section 3.6).

To ensure the construct validity of the risk evaluation tool (SABRE), the following was performed.

- The study used state-of-the-art techniques, e.g., pair programming and code reviews and followed current best practices throughout the implementation, such as optimization of hyperparameters of RFS.

- Thorough white-box testing for verification (including external libraries) was performed.
- Moreover, the author consulted practicing project managers and engineers with many years of relevant professional experience in software engineering and project management disciplines.

To improve the *conclusion validity*, RFS was applied, which is a very robust method and quasi-independent of the interdependence of the risk factors. In addition, this method handles different (discrete or continuous) scales of risk factors. The large-scale simulation ensured that the only variables that were insignificant were those that did not influence survival. Nevertheless, with this distribution, the risk effects can be underestimated if the range of the distribution is narrow. Therefore, a wide range (40% of the most likely value) was applied (see Section 3.5.1).

In this following second part, the case-study related validity threats are examined and determined counteractions as follows:

Content Validity: to ensure that the components, variables, and processes included in the case study and the simulation accurately represent the real-world system, several rounds of alignment with the corporate experts proceeded from the planning to the realization phase, from several roles including developers, testers, competency managers, line managers, project managers and senior management to have a full picture included. The assumptions were reviewed with them like input data, equations, and algorithms to make sure they align with the domain knowledge and experience, compared to the academic literature and previously introduced methods from the academic literature.

Construct Validity: it was evaluated whether the simulation captures the theoretical construct that was intended to be modeled. This involved verifying that the variables, relationships, and mechanisms in the simulation align with established theories and models previously demonstrated.

Criterion Validity: Comparing the results of the case study simulation with established benchmarks, empirical data, or previous studies to determine whether the simulation produces outcomes that are consistent with real-world observations proceeded with the previously introduced IT simulation processes, the overall view is visible in the radiographs in the Figure 4.6 for the IT projects simulation and for the use case in the Figure 5.6 to be able to see the results.

Based on these considerations for the validity, the content is judged as suitable for fulfilling the thesis content expectations.

Chapter 6

Summary and Conclusion

This Chapter contains a summary of the work done in the frame of the dissertation with the final articulation of the Research Theses. The chapter also including the theoretical research activities, the simulation validation work with the Case-study. Also the Implications for practictioners and researchers are articulated in this part. As a highlight of this research, the novelties are summarized with some outlooks of the further future researches.

6.1 Summary

The rapid rise of technology requires a deep understanding and efficient management of software programs or applications, which are essential to modern business operations in industries such as info-communication, automotive, healthcare, aerospace, and many other arts. An unparalleled shift could have been witnessed toward a software-based economy in recent decades. Companies of all sizes use software to innovate, optimize workflows, and offer value to customers. Software creation and maintenance now have a greater economic impact. Software project failures though can hurt the economy. Delays, budget overruns, and poor software quality cost money and reduce market competitiveness. Reducing these risks and maximizing software development project economics requires effective handling of applications throughout their whole lifecycle. Application Lifecycle Management (ALM) offers a framework for such a solution as it manages the entire software application process from inception, via development and maintenance of the application till its retirement it can ensure long-term economic returns for software investments. This is why it is highly important to research this area and provide academic solutions for the business challenges listed above.

- ✓ Research ALM scientific literature for
  - [+] definition and scope identification,
  - [+] enabling definition determination for methodological research,
- ✓ To confirm the applicability of Matrix representation for scheduling investigation, including:
  - [+] simulation (artificial) environment setup,
  - [+] TPM, APM and HPM feasibility check,
  - [+] TPM, APM, and HPM scheduling efficiency analysis.
- ✓ To examine the effects of risk factors on the IT project's structure for scheduling.

As a validation of the above described goals, a relevant ALM case study with scheduling performance evaluation was conducted in an ALM environment by an Automotive Supplier company.

The contribution was threefold in this dissertation. On the first count, there is a contribution to the ALM literature by providing a synthesized ALM definition supporting future methodological research as it is based on a thorough systematic literature review for the definition and modeling of ALM based on peer-reviewed quality academic sources. This step was necessary as ALM is a relatively new and yet less researched area in the scientific literature, with mostly vendor-driven information available in the area. Therefore a rigorous systematic literature review was conducted including as wide a range of sources as possible, with the presumption of keeping the quality, by selecting peer-reviewed sources. It was a keyword search for explicit ALM definition, and after identifying the relevant sources, a critical review was performed to gain the content. It was revealed that overall 7 types of ALM definitions are occurring among them. The most frequently occurring definitions highlight that ALM is strongly related to artifact management during the application management, also that it is a process similar and based on the PLM but specific for SW development, and thirdly, that ALM is a paradigm, a holistic consisting of governance, development and operation/maintenance elements. Based on these relevant sources and field experience, I have proposed a unified ALM definition, that is joining the understanding in the different aspects like scope, phases, key components, scheduling methodologies, flexibility, and metrics. Considering the fact, that such a widespread summary description was not yet available previously in the literature, this definition can serve as a base for future investigations by any scholars to proceed with methodological research by understanding better the scope and attributes of ALM.

On the second count, quantitative research proceeded for applicability and sensitivity checks of known PM methodologies such as traditional, agile and hybrid, to see how efficiently they provide solutions for ALM scheduling problem. The matrix-based scheduling algorithm which is applicable for projects was extended with a flexible schedule handling option in the form of non-planned task handling. The project management execution types were then represented as agents, respectively for Traditional Project Management (TPM) a Traditional Project Management Agent (TPMA) was created, similarly for Agile PM, and APMA, and for Hybrid PM, and HPMA. In the environment then the scheduling performance is evaluated and described how the algorithms are performing. In addition to the performance evaluation also a risk evaluation was proceeded concerning the extended scope of the ALM compared to the classical project scope understandings from the academic literature.

On the third count, a present-day case study is executed in an ALM environment at an automotive supplier company that is facing application development challenges, and after the modeling and evaluations recommendation is provided to their management about the results and potential changes for improvements. The case study is an important pillar, as previously in the theoretical and simulation environment proven methodologies were tested in a real-life problem. The environment and problem definition involved several professionals from the execution level up to management levels in several rounds to ensure the representativity. The simulation with the case study data showed results according to the expectations based on the theoretical concept. The company appreciated the academic support for confirming an efficient way of working determination in their business area.

As an overall summary, it can be stated, that the targeted goal of the dissertation is fulfilled to extend the ALM scientific literature with several value-added results, which also appeared in several conferences, proceedings, and in the form of article publications. Practitioners involved in the process were also highlighting the positive effect by asking and answering questions outside of their daily routine, helping them to rethink the way of their work, and even supporting it with proven academic data. The ALM area though far from being complete, is rather the start of a new journey for potential researchers based on the provided results.

In terms of scheduling, the traditional project management approach and the implemented TPMa operate only in terms of multimode task completions. This approach assumes that tasks can be completed in different kinds of ways. In contrast, agile techniques assume a flexible project structure, where dependencies between tasks can be flexible and lower-priority tasks can be postponed until the next project, but usually, only one completion mode is specified. The results showed that in the case of a flexible project environment, where the flexibility rate is high, this approach can truly produce more feasibility, and in this way, it can make remarkably more projects capable of success than traditional approaches. However, this advantage dissipates when the technology requires strict dependencies.

Hybrid techniques allow both multiple modes and flexible structures, and therefore, it is assumed that this is the supreme technique of project management. This assumption is reinforced by the fact that this technique provides the highest ratio of feasible solutions and the best scheduling performance when only the target function (see Table 6.1) is considered. Based on the proposed database, HPMa provides the most feasible solutions; therefore, ***a software development project is more likely to survive the risk effects if a project plan is managed by a hybrid project management approach***.

Project Management agents Assumptions	TPMa				APMa				HPMa			
Multimodes	X								X			
Flexible dependencies					X				X			
Feasibility	Rank	Stage 1	St 2	St 3	Rank	St 1	St 2	St 3	Rank	St 1	St 2	St 3
$f\%$	3	0.48	0.43	0.34	2	0.50	0.48	0.47	1	0.58	0.53	0.50
Risk effects Uncorrelated case	Rank		$i=2$	$i=3$	Rank		$i=2$	$i=3$	Rank		$i=2$	$i=3$
$\Delta f_{i,i-1}\%$	3		0.90	0.81	1		0.97	0.97	2		0.91	0.95
$\Delta TPT_{i,i-1}\%$	3		1.19	1.31	2		1.04	1.02	1		1.02	1.03
$\Delta TPC_{i,i-1}\%$	3		1.21	1.34	1		0.93	0.97	2		1.03	1.17
$\Delta \overline{TPR}_{i,i-1}\%$	1		1.05	1.10	3		1.11	1.14	2		1.06	1.08
$\Delta TPQ_{i,i-1}\%$	1		0.92	0.89	3		0.87	0.91	2		0.90	0.89
$\Delta TPS_{i,i-1}\%$	1		1.00	1.00	3		0.88	0.90	2		0.93	0.91
Correlated case (r>=0.6)	Rank		$i=2$	$i=3$	Rank		$i=2$	$i=3$	Rank		$i=2$	$i=3$
$\Delta f_{i,i-1}\%$	3		0.68	0.61	2		0.84	0.91	1		0.94	0.93
$\Delta TPT_{i,i-1}\%$	3		1.27	1.39	2		1.12	1.18	1		1.09	1.14
$\Delta TPC_{i,i-1}\%$	3		1.28	1.37	1		1.01	1.10	2		1.14	1.17
$\Delta \overline{TPR}_{i,i-1}\%$	2		1.22	1.31	3		1.24	1.37	1		1.20	1.19
$\Delta TPQ_{i,i-1}\%$	1		0.81	0.71	3		0.77	0.70	2		0.78	0.80
$\Delta TPS_{i,i-1}\%$	1		1.00	1.00	3		0.80	0.91	2		0.87	0.87
Scheduling Performance	Rank	Stage 1	St 2	St 3	Rank	St 1	St 2	St 3	Rank	St 1	St 2	St 3
For target functions ($TPX\%$)	3	0.41	0.34	0.14	2	0.71	0.64	0.59	1	0.83	0.81	0.75
Remaining	Rank	Stage 1	St 2	St 3	Rank	St 1	St 2	St 3	Rank	St 1	St 2	St 3
$TPT\%$	3	0.49	0.38	0.28	1	0.89	0.85	0.78	2	0.69	0.63	0.57
$TPC\%$	3	0.41	0.31	0.21	1	0.81	0.76	0.69	2	0.63	0.53	0.47
$\overline{TPR}\%$	1	0.85	0.74	0.44	3	0.41	0.37	0.27	2	0.61	0.51	0.40
$TPQ\%$	1	0.82	0.72	0.61	3	0.62	0.52	0.42	2	0.72	0.62	0.57
$TPS\%$	1	1.00	1.00	1.00	3	0.61	0.51	0.43	2	0.81	0.70	0.62
Pros vs. cons for stakeholders:	Pros		Cons		Pros		Cons		Pros		Cons	
Customer	High quality Full scope		Longest Lower feasibility		Shortest		Lower quality Less content		Highest feasibility Best schedules		No multipurpose version	
Management	Lower res. dem. in time		Highest cost		Lower cost		Higher res. dem.		Highest feasibility		No multipurpose version	
Developers	Lower res. dem. in time						Higher res. dem.		Best schedules			

FIGURE 6.1: Summary table of results

Currently, the flexible project scheduling algorithms are much less sophisticated than the trade-off methods or the MRCPSP algorithms. For example, there is currently no multipurpose version of agile or hybrid scheduling, and only one target is considered in scheduling and risk mitigation. Table 6.1 shows the ranks in addition to the scheduling and risk mitigation values. The results show that the HPMa does not usually mitigate the risk effects the best. Nevertheless, selecting an adequate project management approach and ensuring project flexibility (see Figures C.5, C.6, C.8, C.7 in Appendix C) are the main factors for both the feasibility and performance of scheduling and mitigation.

Notwithstanding these findings, because of technical requirements, there are substantially more obligatory dependencies between tasks, and the flexible project management approaches do not achieve better performance.

6.2 Research Theses

In this section, I am concluding the research theses based on the research questions (**RQ1-RQ3**) and research assumptions (**RA1-RA3**). An overview table is visible in the Section 6.6 side-by-side listing the research questions, assumptions, and theses.

RT1: Based on the unified ALM definition -including time, cost, resource and quality aspects also- the matrix-based planning model can be applied for scheduling purposes in the ALM environment, as it fulfills the flexible requirements relevant to the planned and non-planned activities.

RT2: The ALM scheduling problem can be set up as an extended project management scheduling problem and the existing project scheduling methodologies (TPM, APM, HPM) provide feasible solutions in the ALM environment with different performance levels:

RT3: A total of 9 risk factors extended from the project scope to the ALM environment are confirmed related to scope, time, cost, resources, and quality; also ALM specific risk factors focusing on the scope change, specific to ALM scheduling were identified:

Related to RT1 further information as an extension is provided for better scope and understanding:

ALM is a holistic approach to managing software applications throughout their entire lifecycle, from inception to retirement. which can be described in the Matrix-planning model also.

ALM is realized by integrating and managing various activities and work products related to 3 ALM functions such as *governance, development and operations, including maintenance*. Governance is an overarching management activity during the whole lifetime of the ALM, however, its importance is higher in the upstream due to its influence factor. Development is mostly related to the classical SW development projects containing the main R&D related work. Operations and maintenance are rather similar to a service. However, the fact that in this phase, next to the bugfixing, additional non-planned tasks can appear in different sizes makes it unique. There are primary 3 main ALM milestones for ALM: *Ideation, Deployment and End-of-life*; and there are 7 phases including *requirements gathering, design, development, testing, deployment, maintenance, and decommissioning*. The ALM core components are for supporting the lifecycle with processes and tools such as *version control, issue tracking, continuous integration, and deployment automation*. These components play a crucial role in scheduling and resource allocation.

This ALM definition can be used to enable a matrix-based project-planning model to represent Application Lifecycle Management problems. It addresses the demands of renewable and non-renewable resources, time, cost, and quality with single and multiple execution modes.

Related to RT2 further information as an extension is provided for better scope and understanding:

For improving the scheduling performance, the handling of the non-planned tasks must be defined in the contractual part already to identify the flexibility in handling and decide which PM approach to utilize accordingly.

Based on the conditions the followings can be proposed to be used:

1. Traditional PM approach: performance with additional tasks planning becomes an incremental model. Multi-mode execution is possible, however, no further priorities can be respected due to the fixed execution order. Senseful to apply in case there is an execution buffer included from the start, else negative effects can be reduced by multi-mode approach only.
2. Agile PM approach: Scheduling on sprint level will not be adapted due to non-planned tasks not being allowed at this level. Only a higher level of planning between the sprints possible to rearrange the next planning session with the assigned priorities.
3. Hybrid PM approach: the most allowing case, multi-mode execution is allowed and priorities can be assigned also the non-planned tasks.
    - A, Non-planned tasks are treated as Change Requests and directly compensated and possible to execute them.
    - B, Non-planned tasks are treated within a frame contract and fulfilled them within those boundaries.

In overall the Hybrid PM approach with traditional elements performs the best.

Related to RT3 further information as extension is provided for better scope and understanding:

By increasing the additional tasks ratio the feasibility and performance behavior is changing. I have identified those factors that are influencing the scheduler's performance in the ALM area also: The following risks are found to be relevant in ALM also: scope creep, changes in requirements, budget overruns, schedule delays, resource constraints, feasibility of problem, and quality issues. Risk factors that appear mostly in the ALM area, like lack of traceability and version control issues, appear due to the unique setup with the non-planned tasks appearance.

So the research theses were defined to answer the research questions, and based on the research assumptions can be confirmed.

6.3 Implications

6.3.1 Implication for practitioners and managers

The primary contribution involves completing a meticulous evaluation of the current body of literature and performing an extensive search to precisely define ALM. This systematic literature review has unique qualities that make it remarkable in its sector. It can also provide valuable insights to professionals from a practical perspective, as no similar review existed prior to this. If the objective is to gather information on the fundamental concept and the extent of coverage for experts, this can be achieved through a theoretical assessment of the dissertation. Furthermore, the inclusion of a comprehensive and critical evaluation in the unified ALM definition can greatly assist the academic community in doing research and expanding their understanding of the ALM field. This, in turn, will also benefit professionals in their job.

Firstly, following the establishment of the unified definition, it is advisable to conduct empirical studies to verify and expand upon the findings of this research. Therefore, both researchers and practitioners are encouraged to explore and share mutually their experiences about the real-world applications of ALM definitions in different contexts. This will help to understand how this definition can be applied in practice, including day-to-day work, and provide valuable insights for refining the definition and enhancing the usability of ALM practices. Such input is welcome from practice towards scholarly sources also.

Secondly, the proposed method compares traditional, agile, and hybrid project management approaches in the view of different kinds of stakeholders. It proposes a meta-network analysis method, which has not been applied in software development projects to date, and has also extended it for the ALM environment. The analysis showed that all methods not only have advantages but also have disadvantages. Most of them are in line with experience, but other methods need a deeper analysis. Similar to experience, traditional project management approaches produced the most infeasible project plans. This result completely matches the Chaos Report's results (SGI, 2019), where waterfall projects, which follow traditional project management approaches, provided three times more failed projects. However, this study also demonstrated that a benefit would occur only if at least 20% of tasks and dependencies were flexible (see Figure 4.5). The lesson which has been learned is that when this requirement cannot be satisfied, the agile project management approach can produce more failed (i.e., infeasible) projects. Due to the project flexibility, the other impressive result is that an agile project management approach usually obtains the shortest and least expensive projects, even though specifying a single implementation mode. However, the expense of this strategy is less content and lower quality. For this reason, it is indeed essential to involve customers for whom the scope of activities to be excluded from the project should be defined (see Table 6.1). At the same time, it is also a vast challenge for developers to manage many parallel activities simultaneously. The hybrid project management approach can take advantage of both flexibility and the choice of completion modes for scheduling; therefore, it provides the best schedules and those that are most feasible, and after the risk analysis, those with the most survived project plans, but these values are best only for the target functions.

The study showed that the most important factor for the feasibility of a project plan is to select an adequate project management approach. The hybrid and, especially, the APMas are better in the flexible project environment. In this case, more feasible and better (i.e., shorter, less expensive, etc.) projects can be specified. Nevertheless, the project structure, such as the size and the parallelization (i_2), are less important factors for survival. Currently, flexible approaches are also used in many non-IT projects. The results showed that the flexible nature of the project rather than the project's specific structure can increase the success of the project or mitigate the risks more. The paper showed that extended meta-network analysis can be used for exploring the effects of flexibility. Agile and traditional project management approaches can usually better mitigate the effects of risk factors, while the hybrid approach helps to ensure the most surviving projects.

Related to the automotive case study available data can become information and knowledge for organizational setup and scheduling for this specific industry. The future for smart actuators and the challenge of SW becoming a product (SWaaP) leading to the Application Lifecycle Management world already a step-by-step reality. Industry must have also input from academia related to process, schedule optimization, organization challenges, and many more.

6.3.2 Implication for researchers

The showed systematic literature review study has discovered uncertainties and contradictions in current definitions found in academic literature, and proposed a widely acknowledged definition for future research. Researchers are encouraged to participate in these standardization activities to promote a more consistent and compatible comprehension of ALM across various businesses. ALM is a multidomain subject that intersects with software engineering, project management, and other areas. With the availability of a clear definition, the next logical progression would involve incorporating ALM concepts with developing technologies like scheduling advancements, artificial intelligence, machine learning, and DevOps. Researchers are encouraged to investigate how the definition of Application Lifecycle Management (ALM) can be utilized to improve for example software development and maintenance processes, boost feasibility, and increase efficiency in ALM duties. It is important to note that the clarified definition of the ALM task has made it easier to provide a clear and transparent description using methodical tools. For instance, a matrix-based representation that allows for in-depth analysis of scheduling issues.

The proposed multi-layer network analysis and survival analysis-based risk evaluation (SABRE) tool showed that these techniques can be used not only in construction projects but also in software development projects. With SABRE, the study showed that agile and traditional project management approaches are more sensitive if risk factors are correlated with each other (see Table 6.1). The proposed simulation model can investigate the impact of formerly not or hardly studied risk factors, such as project structures, shocks, and flexibility. In addition, with the proposed model, scholars can dynamically tune the level of flexibility in hybrid and agile approaches. Further kinds of risk factors and their interdependencies can be easily added to the existing networks to enhance simulation models.

The study also highlighted an important shortcoming of agile and hybrid approaches, namely, that they have no multipurpose version that can balance the different kinds of goals of stakeholders.

Another possible extension of the proposed model, as yet hardly studied, is to examine flexible multilevel project risk management, where the risk effects of simultaneous projects may also impact each other.

6.4 Novelty of research and Limitations

This section highlights the most significant contributions of the dissertation research in the context of existing academic literature.

First needs to be noted that the ALM-related literature is still scarce and expects growth from several perspectives, due to the fact that it started up as mostly vendor-driven, and not even a clear or unanimously accepted is existing for ALM definition. This is due to the vendor's purpose to form the ALM according to their business interest and also to the fact of the quick development of the concept itself. Business-related authors and professionals are sharing and contributing to the general knowledge base of the ALM, however, the scientific community has currently limited time and efforts invested in the area. Thus, the dissertation's first parts focused on the literature review, in a broader sense to get to know the ALM more in detail, and more focused on finding existing ALM definitions so that as a next step a unified concept can be created to support further methodological researches by the academic community which is under-researched today. The cross-sectional systematic literature review method was used to provide the base for the existing definitions in a wide scope of academic literature. Then a critical review proceeded to analyze and create a unified ALM definition intended to integrate the scopes and attributes. So the first significant contribution was the created systematic literature review on the ALM definition. By default, the SLR is a contribution as none existed before. This can be used also as a base for a longitudinal or a meta-research, e.g., for SIMILAR method (Z. Kosztyán, Csizmadia, et al., 2021) for further extending the ALM literature. The additionally proposed ALM definition can be a base for further research by academics, opening up new horizons for methodological research, as the problem already exists in the business, as revealed by the case also. Limitations related to this systematic review though exist which need to be highlighted. For the cross-sectional reviews relevant such as capturing the information only in a specific time available, which was here in the research part, however, the longitudinal perspective and historical contexts are not clearly and precisely available. The above limitation with the quoted SIMILAR method could be compensated in the future. The other limiting factor for the systematic literature review is shown by the setup already, with the limitation of the input for example the language, quality of sources, and naming differences for ALM existing in the early years. An exploratory study to resolve these would be beneficial to proceed also as the next steps in the research field. These limitations must be considered when interpreting the findings and conclusions drawn from the ALM definition review also.

Secondly, a matrix-based method was developed and proposed to examine the feasibility of IT projects with existing project management approaches (TPM, APM, HPM) programmed as agents. Similar feasibility-related comparisons did not exist before in the academic literature based on such complex simulations using real-life data as input. The limitation of the available data sources (IT relevant projects), however, still no clear and ALM-specific database sources are available in the academic literature. hopefully, in some years such information sources will be available by experts. The method and setup using the simulations also contain several limitations and from an interdisciplinary point of view a simplified model only focuses clearly on well-defined ranges, which were shown in the literature review. Limitations related to the success understanding also now very basically understood along the iron triangle, assigning score values for fulfilling constraints. A different, more mature model, or understanding might be extended to reflect the stakeholder satisfaction for example. Therefore the second main contribution is also to be considered to be limited from these perspectives.

Then for the third point, a case study was proceeded with an automotive supplier company ALM-related challenging situation evaluation and using the previously demonstrated matrix-based method extended to the existing ALM environment. The case study involved several experts, and managers in a leading automotive supplier that had not yet recorded such a complex HW-SW-related approach in the literature beforehand. The Application Lifecycle Management scheduling problem was recognized and realized after the interviews and internal investigations followed up with leading managers. The quantified data and scheduling problem analysis with several approaches (TPM, APM, HPM) revealed deeper context and potential further organization development for the company towards higher efficiency. Limitations related to the simulation exist here also. Related to the case and industry specifics are also limiting the ALM scope, further industries and setups are also to be examined to have a broader view on the ALM realizations.

As today more and more applications are developed by private and public sectors, the need for this specific management, i.e., Application Lifecycle Management is getting more and more into the focus both by professionals and academics. In the first decades, the adaptation of some ALM concepts is already a task for organizations, the next step will be the efficiency increase, for these entities must rely on academic inputs also, e.g., scheduling methodologies and tailoring processes for their fitting needs.

ALM is in understanding clarified and defined in the first part of the dissertation. However, for the methodological research part for scheduling, limitations were also applied for applicability and scope management within the dissertation limitations.

The project management agents' usage for matrix structure applicability and the case study have a strong connection, and for first realization proves the feasibility. However, this is not excluding but rather inviting further representations to be invented and elaborated by the scientific community for the ALM field. De-limitations for the matrix-type problem description are only one way of solution for the ALM scheduling problem.

Simulations limitation for scheduling is present for the model. For example for the Agile model, the premises already contain the project-related tasks in a fixed form from the first iteration, which means the additional definition is not considered. Thus only static scheduling is possible. The changed tasks within the Agile project run have not handled yet. A potential future solution can be an online scheduler for the iterations to be able to have dynamic scheduling enabled. Due to simulation restrictions and more areas to involve from real life to test it and have a broader perspective.

In order to facilitate the comparison between simulated and real-life projects, the projects were primarily evaluated based on their time and renewable resource requirements. Nevertheless, the presence of nonrenewable materials, the associated costs, and the need for high quality would create opportunities for additional research. Furthermore, the comparison of existing databases with the newly introduced artificial and real-life application lifecycle projects, agile application projects, and agile multiproject databases would be intriguing. At present, there are no existing databases that contain real-life application project data with the ALM approach. As a result, the ability to directly compare simulated (fake) data with real data is restricted to individual projects only. Introducing ALM plans from other industries would enhance the breadth of research.

6.5 Conclusion and Future research

The main goal of the dissertation was to provide a thorough, meaningful, and practical evaluation of the recently emerged field, the Application Lifecycle Management. The research provides new insights into important aspects of the understanding and base for future methodological studies.

Firstly, the ALM field was thoroughly researched by a systematic and critical review using the existing scientific literature available in the area. Highlighting that also that the ALM is a specific scope, where next to the academic literature the business, vendor-driven literature is playing a decisive role in the development of the context. Based on the available academic literature created a unified ALM definition to support future methodological research, which did not yet exist in the field.

Secondly, the ALM characteristics were structured in a matrix representation form, for that also methodological research was conducted for scheduling efficiency for project management approaches like Traditional-, Agile-, and Hybrid project management and their risk examination. Such kind of evaluation of scheduling methodologies was not yet present in the literature, to be able to see and determine how the different methodologies are fitting to different structures.

Thirdly, a recent company problem was modeled in a case study with the ALM problem in the Automotive supplier industry, where the results for the project management approaches were also examined within a simulated comparison, then provided recommendations to the company experts and management.

For future research then the base for comparisons and evaluations is available, opening up new doors for either further tuning of the hybrid methodology as a potential way. However, as mentioned also further SW-related methodologies can be analyzed and evaluated, and potential application development for specific areas can result in further improvement options. In summary, the next steps for methodological optimizations are established and encouraged for academia and field experts.

6.6 Research summary table

See the below Table contains the summarized Research Questions, Assumptions and Theses for a better overview combined together.

Item	Statement
RQ1:	*How can a planning model be identified based on available scientific literature definitions that represents the Application Lifecycle Management (ALM) scheduling problem?*
RA1:	A planning model can be identified based on the unified ALM definitions from the scientific literature.
RT1:	Based on the unified ALM definition -including time, cost, resource and quality aspects also- the matrix-based planning model can be applied for scheduling purposes in the ALM environment, as it fulfills the flexible requirements relevant to the planned and non-planned activities.
RQ2:	*Do the existing project management scheduling methodologies (TPM, APM, HPM) produce feasible solutions in the ALM environment, and how are they performing?*
RA2:	The TPM, APM, HPM project management approaches using the matrix-based planning method can be extended to solve the scheduling problem, and result in feasible solutions with different results in the ALM environment. A simulation framework can be constructed to handle flexible dependencies and non-planned tasks.
RT2:	The ALM scheduling problem can be set up as an extended project management scheduling problem and the existing project scheduling methodologies (TPM, APM, HPM) provide feasible solutions in the ALM environment with different performance levels:
RT2.1:	TPM approach manages more feasible projects in case the flexibility is lower, and it provides all cases the highest score on customer satisfaction due to execution of all defined tasks, and fewer resources per time unit, however also the highest project budgets with longest projects due to the worst project scheduling performance.
RT2.2:	APM approach manages significantly more feasible project than TPM when the flexibility is higher, and in general the shortest projects with the lowest budget if other target function is selected, however requires more resources per time unit than TPM.
RT2.3:	HPM approach provides the most feasible projects in case higher flexibility is present, and shows the best performance for targets to reach (cost, time, quality), and this secures the best total project value.
RQ3:	*What are the risk factors for the scheduling problem in the ALM environment, and how are they influencing the feasibility and scheduling performance?*
RA3:	There are existing project-related risk factors that can be extended for ALM scheduling problems, however, due to the differences between project and ALM scope, ALM-specific risks appear also, which can have an effect on resources, cost, and timing, and can influence the feasibility and scheduling performance.
RT3:	A total of 9 risk factors extended from the project scope to the ALM environment are confirmed related to scope, time, cost, resources, and quality; also ALM specific risk factors focusing on the scope change, specific to ALM scheduling were identified:
RT3.1:	The 3 most influential risk factors in the ALM environment are the following: the applied project management approach, the degree of structural flexibility, and the correlation between the risk factors.
RT3.2:	In the ALM environment, the low-level risks (changes in cost, time, resources) have a higher impact than the high-level risks (fulfillment of constraints by the target function).
RT3.3:	TPM is the most sensitive to the shock effects, with only small changes (10%) of task demands can result even 10 times higher modifications in the duration, resources, and costs.

TABLE 6.1: Summary table for Research Questions, Assumptions and Theses

Appendix A

ALM definition occurrence classification from relevant literature sources

This appendix contains the Application Lifecycle Management definitions from the Systematic Literature Review process. The key findings for the identified classification sources are marked with the assigned authors. The table represents the following information:

- Column A: Position of the filtered result from the sources list.
- Column B: Authors used the identified definition
- Column C: Publication year of the entry
- Column D: Ranking of the source (Journal or Conference)
- Column E: Definition "Type A": ALM is a process for SW PLM/SDLC.
- Column F: Definition "Type B": ALM is SW development AND maintenance.
- Column G: Definition "Type C": ALM is artefact management tool for SDLC.
- Column H: Definition "Type D": ALM is an SDLC extended with phases after development.
- Column I: Definition "Type E": ALM is a paradigm: governance, development, operation/maintenance.
- Column J: Definition "Type F": ALM is a service for after development part only.
- Column K: Definition "Type G": ALM for quality ensurance.

#	Author(s)	Year	Ranking	A	B	C	D	E	F	G
3	Tüzün, Eray; Tekinerdogan, Bedir; Macit, Yagup; İnce, Kürşat;	2019	Q1					x		
24	Manuel, Bertha Alan; Karnalim, Oscar;	2018	Q1		x					
82	Ragan, Tracy;	2006	Q1							x
167	Ebert, Christof;	2013	Q1	x						
35	Kozma, Dániel; Varga, Pál; Larrinaga, Felix;	2021	Q2				x			
37	Deuter, Andreas; Imort, Sebastian;	2021	Q2	x						
39	Gatrell, Matt;	2016	Q2					x		
41	Henttonen, Katja; Kääriäinen, Jukka; Kylmäaho, Jani;	2017	Q2					x		
50	Bricogne, Matthieu; Rivest, Louis; Troussier, Nadège; Eynard, Benoît;	2014	Q2			x				
78	Erata, Ferhat; Challenger, Moharram; Tekinerdogan, Bedir; Monceaux, Anne; Tüzün, Eray; Kardas, Geylani;	2017	A1					x		
61	Hellerstein, Joseph L;	2008	A2			x				
77	Steinberger, Michal; Reinhartz-Berger, Iris;	2016	A2				x			
121	Steinberger, Michal; Reinhartz-Berger, Iris; Tomer, Amir;	2016	A2	x						
134	Elaasar, Maged; Neal, Adam;	2013	A2					x		
7	Lacheiner, Hermann; Ramler, Rudolf;	2011	B1	x		x		x		
25	Gomede, Everton; Da Silva, Rafael Thiago; de Barros, Rodolfo Miranda;	2015	B1	x						
36	Hallerstede, Stefan; Bullinger, Angelika; Möslein, Kathrin;	2012	B1				x			
40	De Simone, Vincenzo; Amalfitano, Domenico; Fasolino, Anna Rita;	2018	B1	x						
138	Wolvers, Ronald; Seceleanu, Tiberiu;	2013	B1					x		
294	Mordinyi, Richard; Moser, Thomas; Biffl, Stefan; Dhungana, Deepak;	2011	B1				x			
13	Jwo, Jung-Sing; Hsu, Tien-Song; Cheng, Yu Chin;	2013	Q3				x			
16	Kaariainen, Jukka; Valimaki, Antti;	2011	Q3			x				
18	Amalfitano, Domenico; De Simone, Vincenzo; Scala, Stefano; Fasolino, Anna Rita;	2020	Q3			x				
79	Üsfekes, Çağdaş; Tüzün, Eray; Yılmaz, Murat; Macit, Yagup; Clarke, Paul;	2019	Q3					x		
88	Amalfitano, Domenico; De Simone, Vincenzo; Maietta, Raffaele Rodolfo; Scala, Stefano; Fasolino, Anna Rita;	2019	Q3	x						
254	Szlęzak, Paweł;	2014	Q3				x			
274	Seceleanu, Cristina; Johansson, Morgan; Suryadevara, Jagadish; Sapienza, Gaetana; Seceleanu, Tiberiu; Ellevseth, Stein-Erik; Pettersson, Paul;	2017	Q3			x				
42	Bradbury, Dan;	2008	Q4			x				
4	Kääriäinen, Jukka; Välimäki, Antti;	2009	B2			x				
11	Üsfekes, Çağdaş; Yilmaz, Murat; Tuzun, Eray; Clarke, Paul M; O'Connor, Rory V;	2017	B2					x		
17	Akgun, Zuleyha; Yilmaz, Murat; Clarke, Paul;	2020	B2	x		x		x		x
22	Ramler, Rudolf; Lacheiner, Hermann; Kern, Albin;	2012	B2		x	x				
30	Ardiç, Bariş; Tüzün, Eray;	2020	B2					x		
38	Klespitz, József; Bíró, Miklós; Kovács, Levente;	2016	B2					x		
54	Kim, Jeong Ah; Choi, Seung-Yong; Hwang, Sun-Myung;	2011	B2			x				
87	Biró, Miklós; Klespitz, József; Gmeiner, Johannes; Illibauer, Christa; Kovács, Levente;	2016	B2					x		
131	Biró, Miklós; Kossak, Felix; Klespitz, József; Kovács, Levente;	2017	B2					x		
135	Pesola, Jukka-Pekka; Tanner, Hannu; Eskeli, Juho; Parviainen, Paivi; Bendas, Dan;	2011	B2			x				
166	Wendel, Heinrich; Kunde, Markus; Schreiber, Andreas;	2010	B2			x				
334	Reinhardt, Wolfgang;	2009	B2	x						
2	Kääriäinen, Jukka; Välimäki, Antti;	2008	B3			x				
153	Pesola, J-P; Eskeli, Juho; Parviainen, Päivi; Kommeren, Rob; Gramza, M;	2008	B3			x				
189	Troubitsyna, Elena;	2019	B3			x				
19	Klespitz, József; Bíró, Miklós; Kovács, Levente;	2016	B4							x
26	Jwo, Jung-Sing; Cheng, Yu Chin; Hsu, Tien-Song; Liu, Chun Hsin;	2010	B4				x			
49	Pekšēns, Ivo;	2013	B4			x				
108	Herden, Sebastian; Zwanziger, André; Robinson, Philip;	2010	B4				x			
137	Oberhauser, Roy; Schmidt, Rainer;	2007	B4				x			
64	Amalfitano, Domenico; De Simone, Vincenzo; Fasolino, Anna Rita; Scala, Stefano;	2017	B5	x	x	x	x	x		x
9	Kääriäinen, Jukka;	2011	Dissertation			x				
94	Samra, Taranjit Singh;	2012	Dissertation				x	x		
262	de Almeida Calheiros, Giovanni;	2019	Dissertation				x			
5	Rossberg, Joachim; Olausson, Mathias;	2012	Book		x					
8	Rossberg, Joachim;	2008	Book		x					
12	Aiello, Bob; Sachs, Leslie;	2016	Book			x				
27	Arya, Anjali; Böhm, Markus; Bose, Bhaswar; Cerveau, Laurent; Endholz, Petra; Geier, Freddie; Krause, Maximo Romero; Krcmar, Helmut; Leimeister, Stefanie; Madhukar, Irvathraya;	2011	Book						x	
76	Eigner, Martin;	2021	Book		x					
109	Hundhausen, Richard;	2012	Book	x						
6	Gunnarsson, Ásgeir; Johnson, Michael;	2020	Book Chapter	x						
10	Chanda, Sandeep; Foggon, Damien;	2013	Book Chapter	x						x
20	Olausson, Mathias; Rossberg, Joachim; Ehn, Jakob; Sköld, Mattias;	2013	Book Chapter	x						
32	Scott, John; Buytaert, Nick; Cannell, Karen; D'Souza, Martin; Gault, Doug; Gielis, Dimitri; Hartman, Roel; Kubicek, Denes; Mattamal, Raj; McGhan, Dan;	2015	Book Chapter	x						
34	Zamazal, Klaus; Denger, Andrea;	2020	Book Chapter					x		
45	Rossberg, Joachim;	2014	Book Chapter					x		
56	Hallerstede, Stefan H;	2013	Book Chapter				x			
65	Deuter, Andreas; Otte, Andreas; Ebert, Marcel; Possel-Dölken, Frank;	2019	Book Chapter			x				
74	Cummins, Stephen;	2011	Book Chapter					x		
96	Moreira, Mario E;	2013	Book Chapter			x				
141	Ritchie, Stephen D;	2011	Book Chapter	x						
180	Eigner, Martin;	2018	Book Chapter	x						
190	Amsden, Jim; Speicher, S;	2021	Book Chapter	x						
207	Eigner, Martin;	2021	Book Chapter		x					
272	Wright, Steve; Erkes, Corey;	2021	Book Chapter	x						
288	Denger, Dirk; Herschmann, Otto-Wilhelm; Barisic, André;	2021	Book Chapter					x		
292	Oka, Dennis Kengo;	2020	Book Chapter	x						
332	Dunphy, George; Moukhnitski, Sergei; Kaufman, Stephen; Kelcey, Peter; Campos, Harold; Peterson, David;	2009	Book Chapter	x						

TABLE A.1: Table for the ALM definition for all the included sources

Appendix B

ADM - Application Lifecycle Domain Map

This appendix contains the Application Lifecycle Domain Map analogous to the PDM (Project Domain Matrix) description. The five interconnected platforms are represented in a sequential flow. Due to size limitations, the split is done platform-wise on each page.

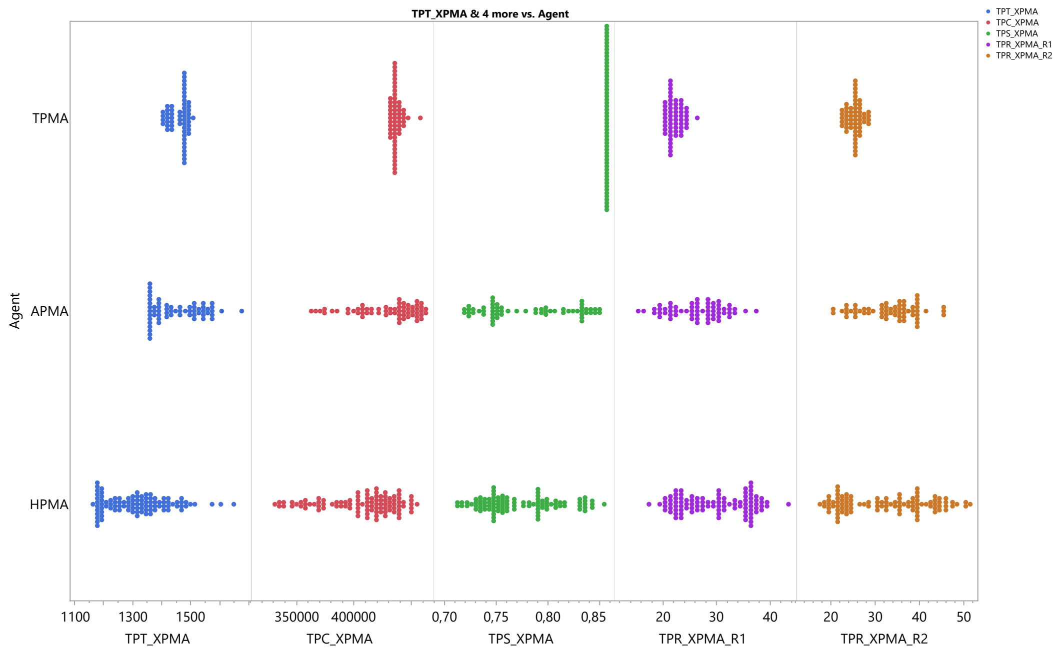


FIGURE B.1: Distribution of feasible solutions of agents and their objectives

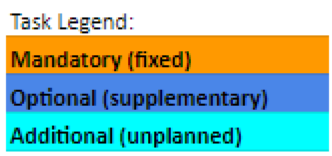


FIGURE B.2: Application Lifecycle Management Domain Map - Task legend

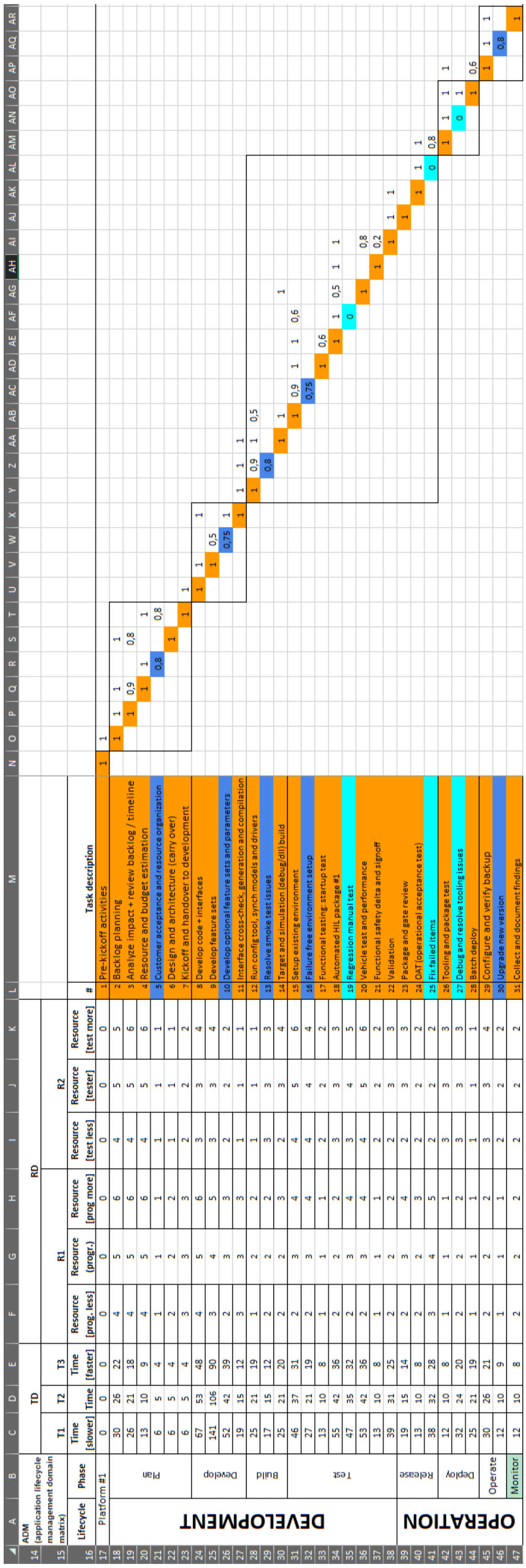


FIGURE B.3: Application Lifecycle Management Domain Map - Platform #1

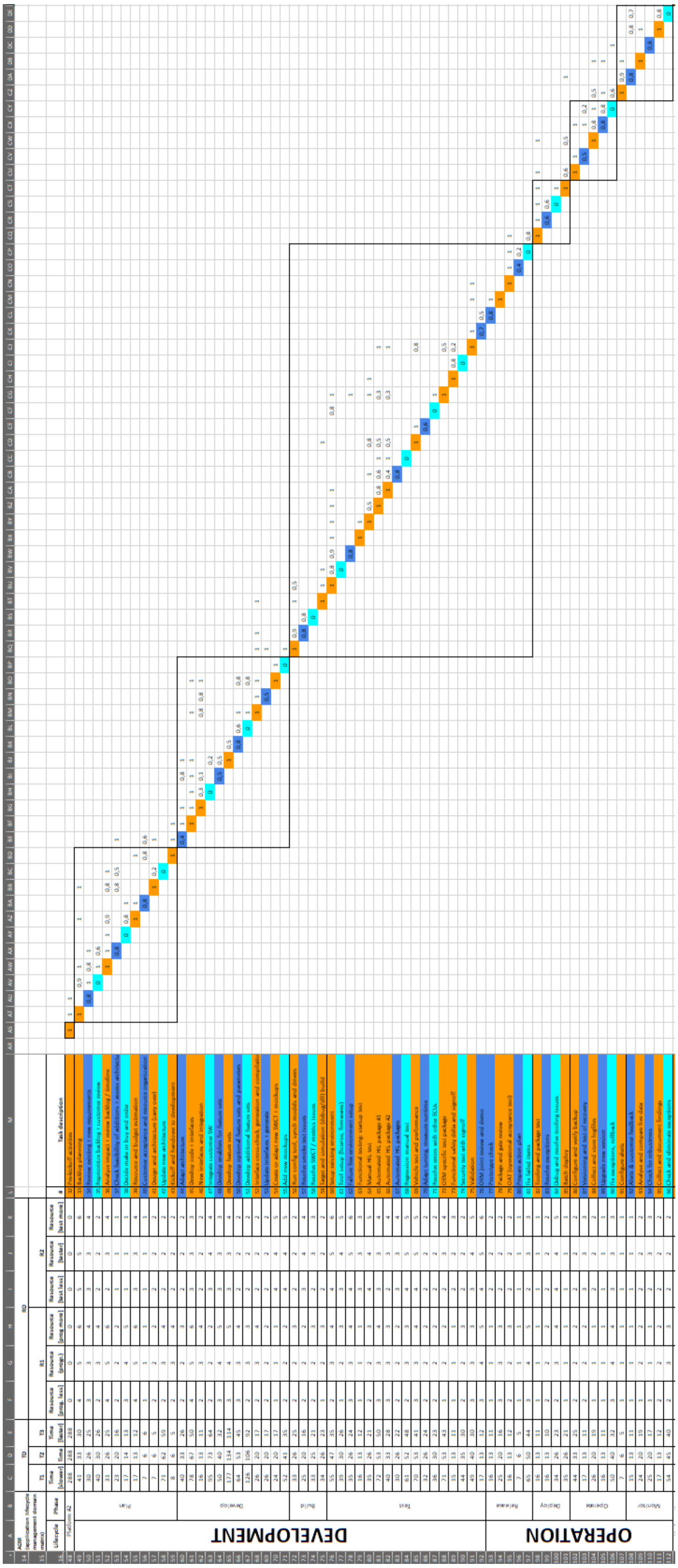


FIGURE B.4: Application Lifecycle Management Domain Map - Platform #2

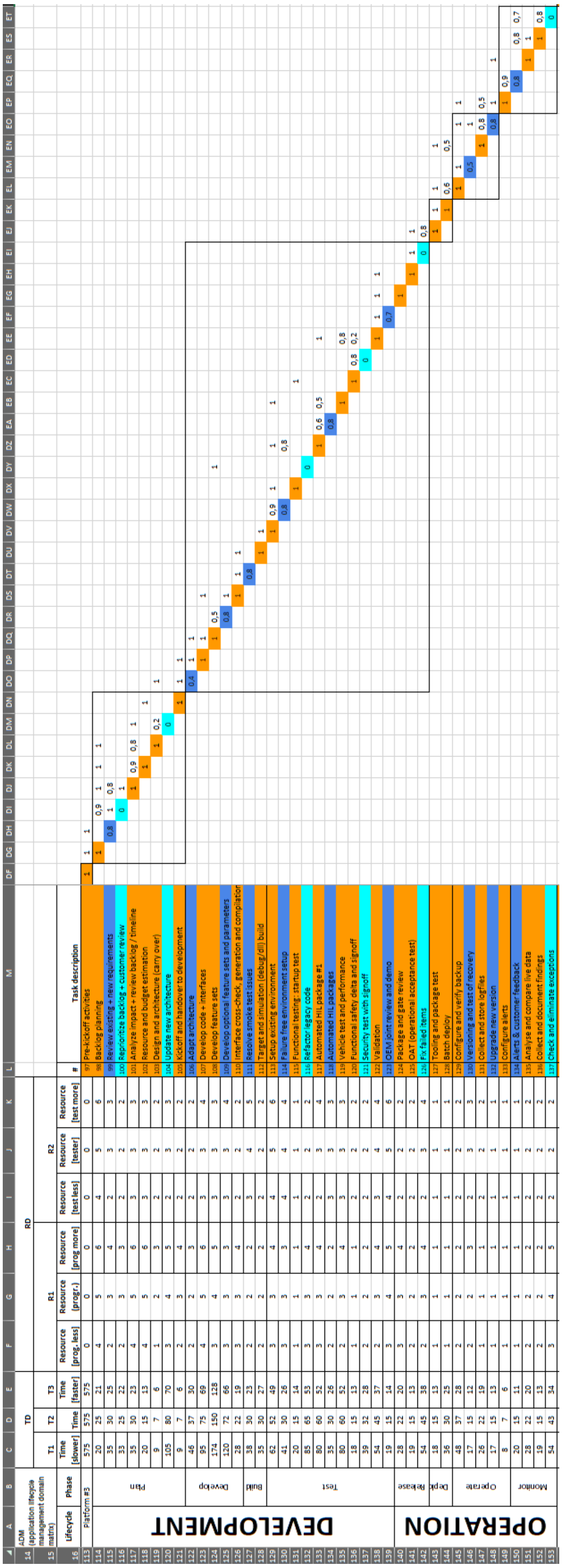
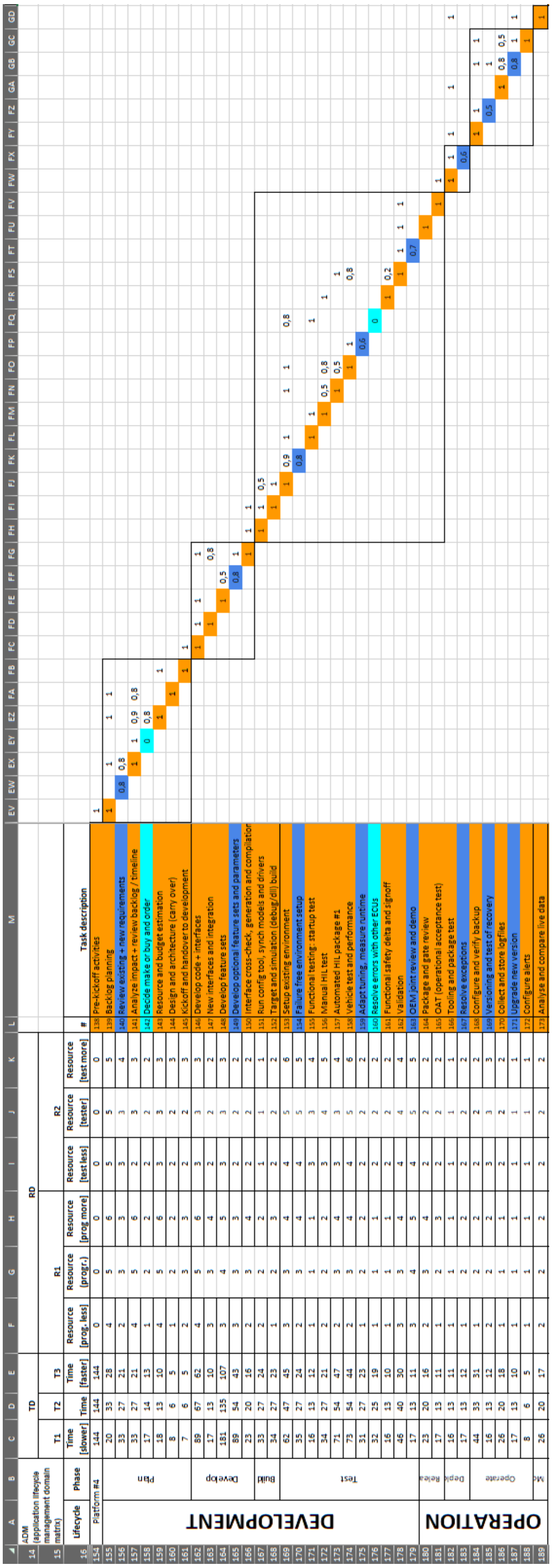


FIGURE B.5: Application Lifecycle Management Domain Map - Platform #3



ADM (application lifecycle management domain matrix)		TD			RD							
		T1	T2	T3	R1			R2				
Lifecycle	Phase	Time [slower]	Time	Time [faster]	Resource [prog. less]	Resource (progr.)	Resource [prog more]	Resource [test less]	Resource [tester]	Resource [test more]	#	Task description
Platform #4		144	144	144	0	0	0	0	0	0	138	Pre-kickoff activities
DEVELOPMENT	Plan	20	33	28	4	5	6	5	5	5	139	Backlog planning
DEVELOPMENT	Plan	33	27	21	2	3	3	3	3	4	140	Review existing + new requirements
DEVELOPMENT	Plan	33	27	21	4	5	6	2	3	3	141	Analyze impact + review backlog / timeline
DEVELOPMENT	Plan	17	14	13	1	2	2	2	2	2	142	Decide make or buy and order
DEVELOPMENT	Plan	18	13	10	4	5	6	3	3	3	143	Resource and budget estimation
DEVELOPMENT	Plan	8	6	5	1	2	2	2	2	3	144	Design and architecture (carry over)
DEVELOPMENT	Plan	7	6	5	2	3	3	2	2	3	145	Kickoff and handover to development
DEVELOPMENT	Develop	89	67	62	4	5	6	3	3	3	146	Develop code + interfaces
DEVELOPMENT	Develop	17	13	10	3	3	4	2	2	2	147	New interfaces and integration
DEVELOPMENT	Develop	181	135	107	3	4	5	3	3	3	148	Develop feature sets
DEVELOPMENT	Develop	89	54	43	3	3	3	2	2	2	149	Develop optional feature sets and parameters
DEVELOPMENT	Develop	23	20	16	2	3	4	2	2	2	150	Interface cross-check, generation and compilation
DEVELOPMENT	Build	33	27	24	2	2	2	1	1	1	151	Run config tool, synch models and drivers
DEVELOPMENT	Build	34	27	23	1	2	3	2	2	2	152	Target and simulation (debug/dll) build
DEVELOPMENT	Test	62	47	45	3	3	4	4	5	6	153	Setup existing environment
DEVELOPMENT	Test	35	27	24	2	3	4	4	5	5	154	Failure free environment setup
DEVELOPMENT	Test	16	13	12	1	1	1	3	3	4	155	Functional testing: startup test
DEVELOPMENT	Test	34	27	21	2	2	2	3	4	5	156	Manual HIL test
DEVELOPMENT	Test	71	54	47	2	3	4	3	3	4	157	Automated HIL package #1
DEVELOPMENT	Test	73	54	44	2	3	4	4	5	6	158	Vehicle test and performance
DEVELOPMENT	Test	31	27	23	1	2	2	2	2	2	159	Adapt tuning, measure runtime
DEVELOPMENT	Test	32	25	19	1	1	1	2	2	2	160	Resolve errors with other ECUs
DEVELOPMENT	Test	16	13	10	1	1	1	2	2	2	161	Functional safety delta and signoff
DEVELOPMENT	Test	46	40	30	3	3	4	4	4	4	162	Validation
DEVELOPMENT	Test	17	13	11	3	4	5	4	5	5	163	OEM joint review and demo
OPERATION	Relea	23	20	16	2	3	4	2	2	2	164	Package and gate review
OPERATION	Relea	17	13	11	1	2	3	2	2	2	165	OAT (operational acceptance test)
OPERATION	Deplo	16	13	11	1	1	1	1	1	1	166	Tooling and package test
OPERATION	Deplo	17	13	12	1	2	2	2	2	2	167	Resolve exceptions
OPERATION	Operate	44	33	31	1	2	2	2	2	2	168	Configure and verify backup
OPERATION	Operate	16	13	12	2	2	2	3	3	3	169	Versioning and test of recovery
OPERATION	Operate	26	20	18	1	1	1	2	2	2	170	Collect and store logfiles
OPERATION	Operate	17	13	10	1	1	1	1	1	1	171	Upgrade new version
OPERATION	Operate	8	6	5	1	1	1	1	1	1	172	Configure alerts
OPERATION	Mo	26	20	17	2	2	2	2	2	2	173	Analyse and compare live data

FIGURE B.6: Application Lifecycle Management Domain Map - Platform #4

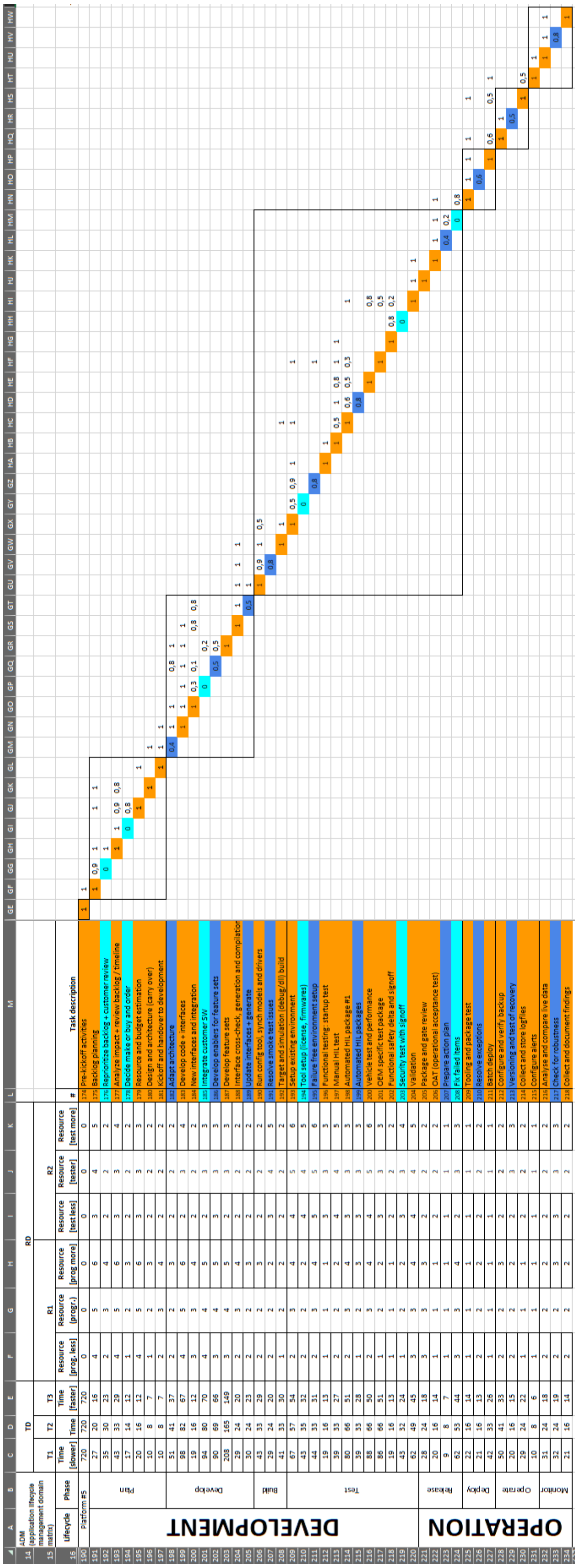


FIGURE B.7: Application Lifecycle Management Domain Map - Platform #5

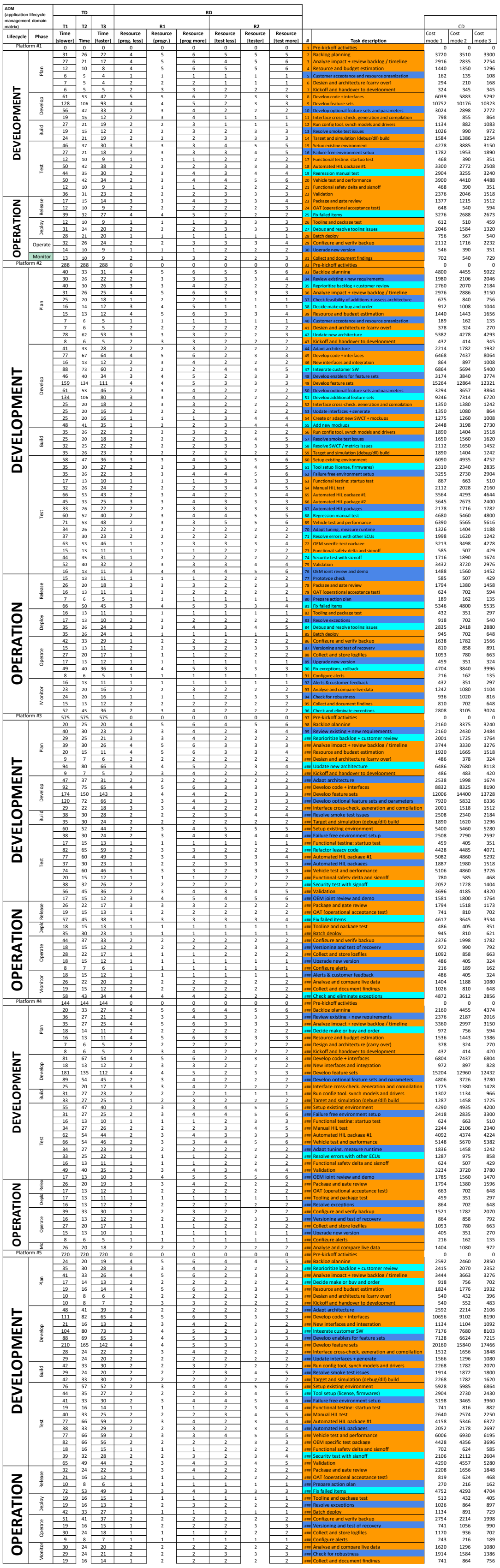


FIGURE B.8: ADM representation highlighting the information for completion modes: Time-, Cost- and Resource Domains (not the simulation model)

Appendix C

Supplementary statistical analysis

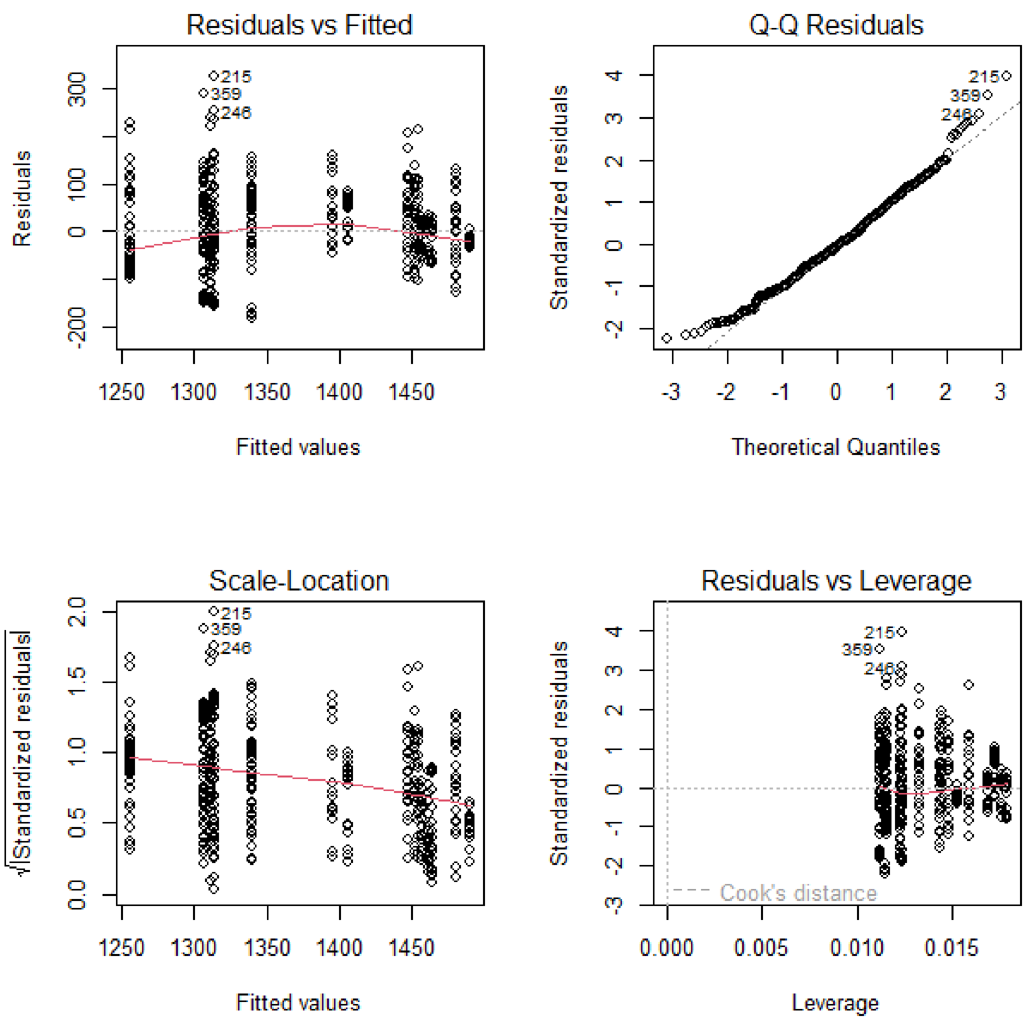


FIGURE C.1: Diagnostic plots for durations (time)

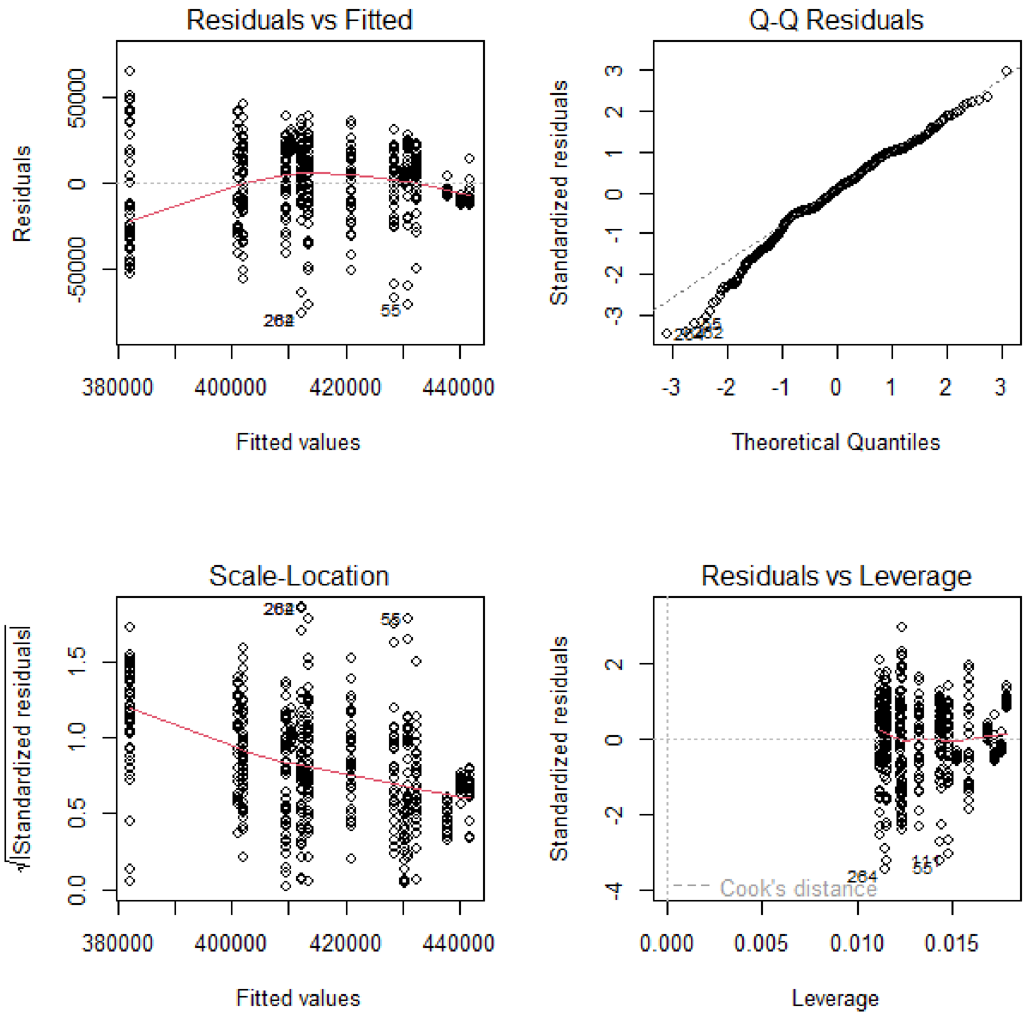


FIGURE C.2: Diagnostic plots for budgets (cost)

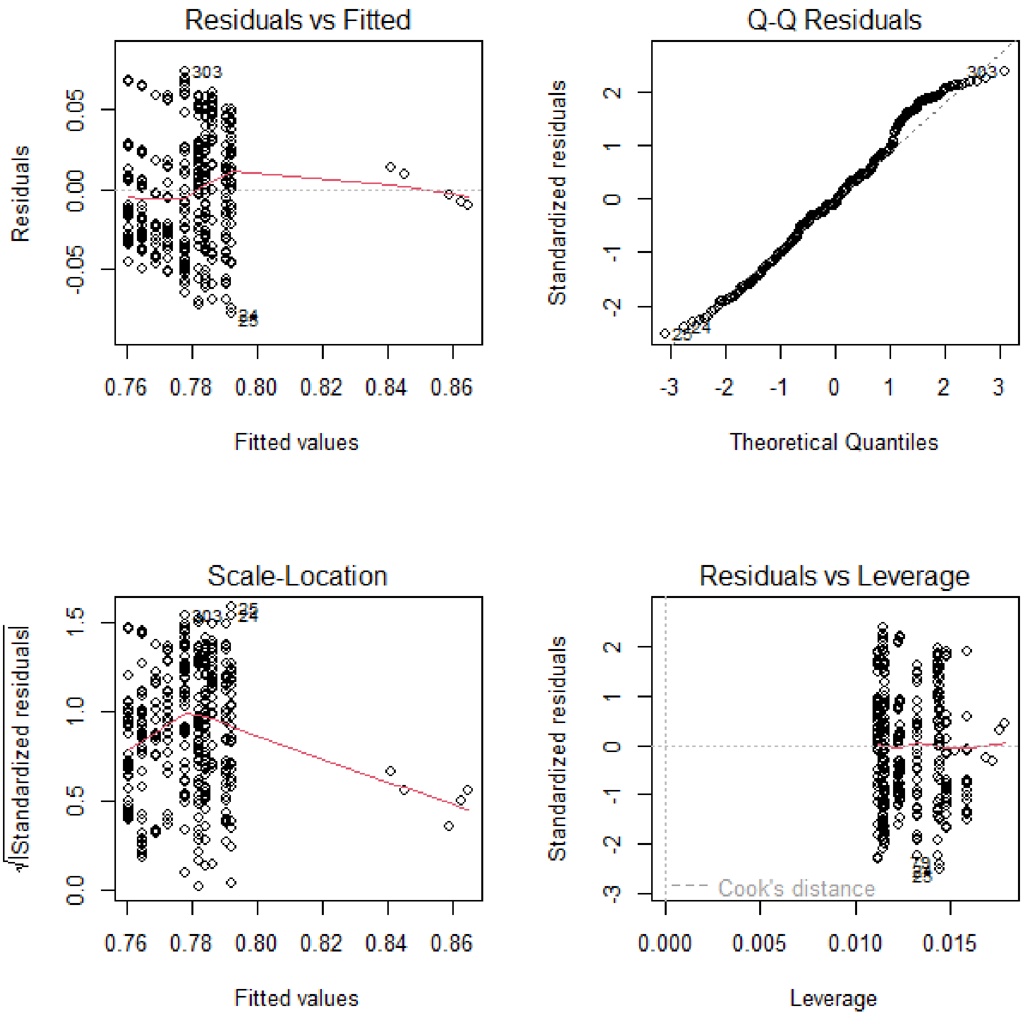


FIGURE C.3: Diagnostic plots for scores (scope)

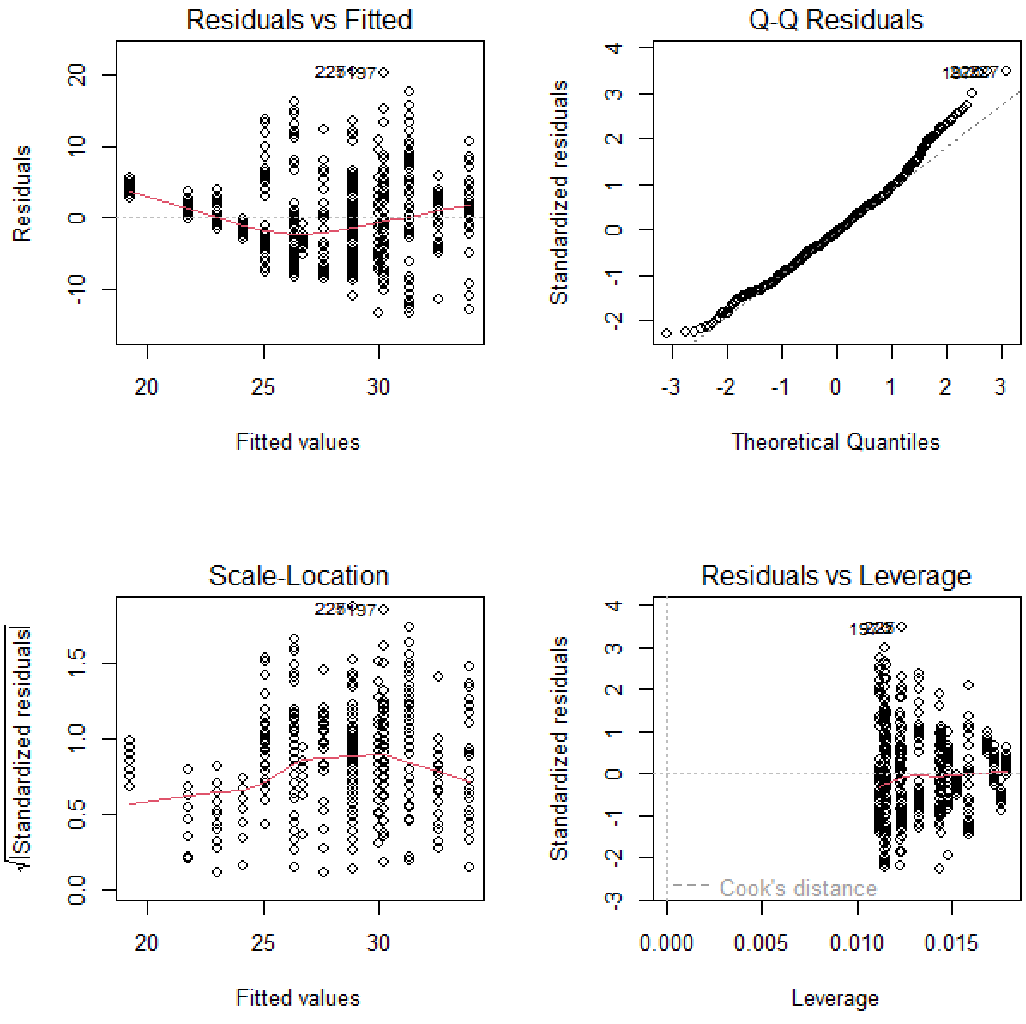


FIGURE C.4: Diagnostic plots for renewable resources

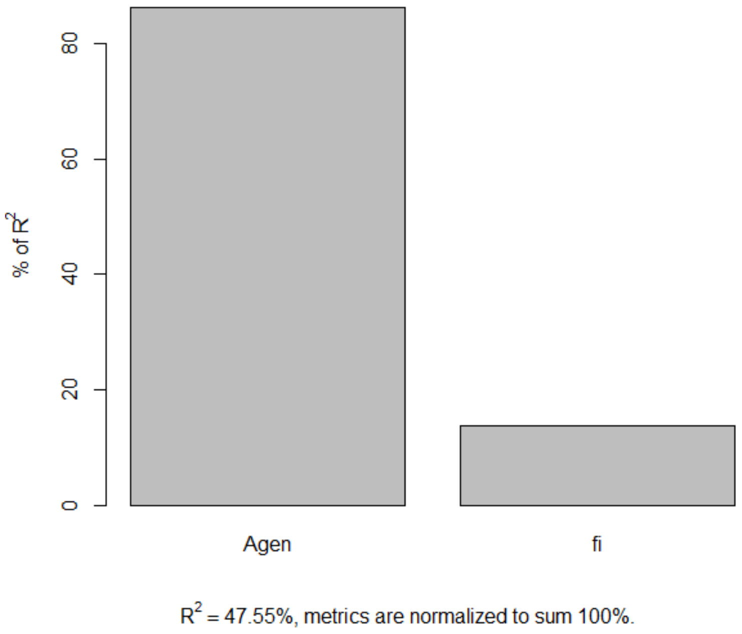


FIGURE C.5: Relative importance of agents and objective functions for durations (time)

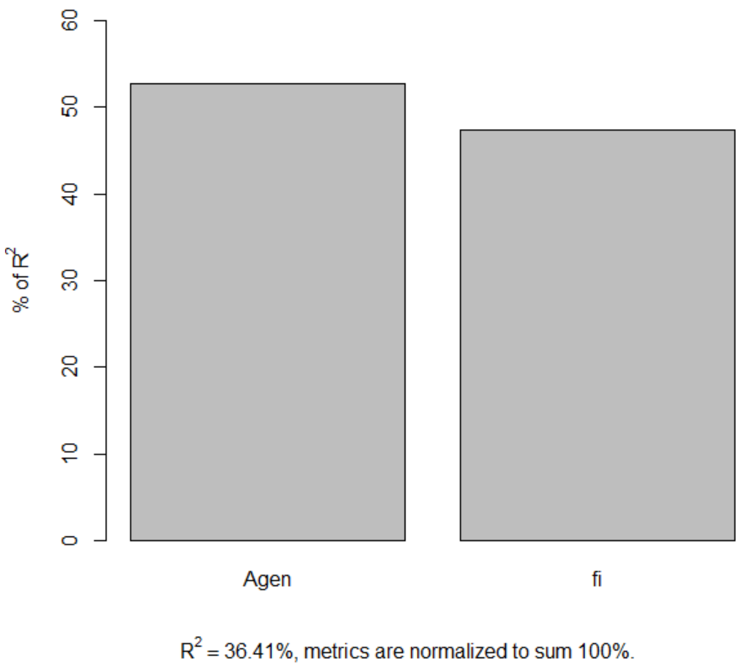


FIGURE C.6: Relative importance of agents and objective functions for budget (cost)

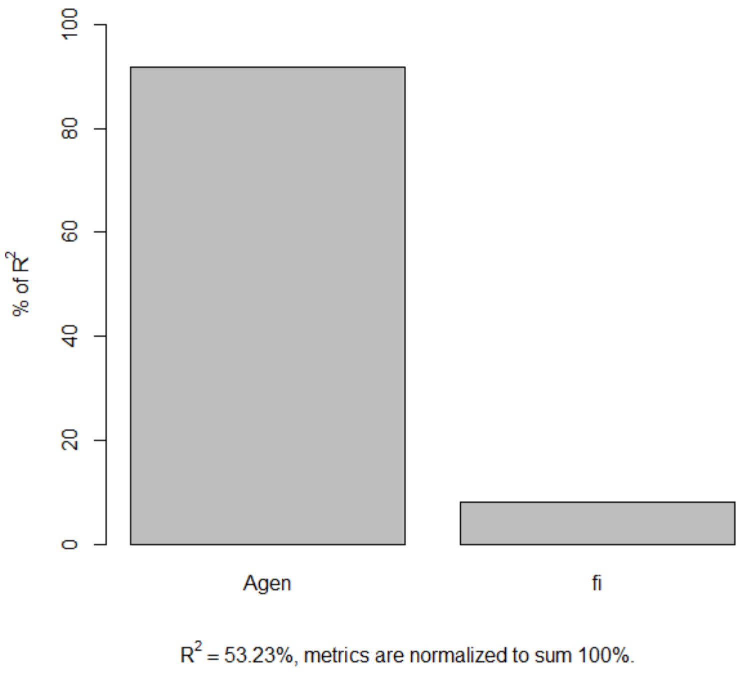


FIGURE C.7: Relative importance of agents and objective functions for scores

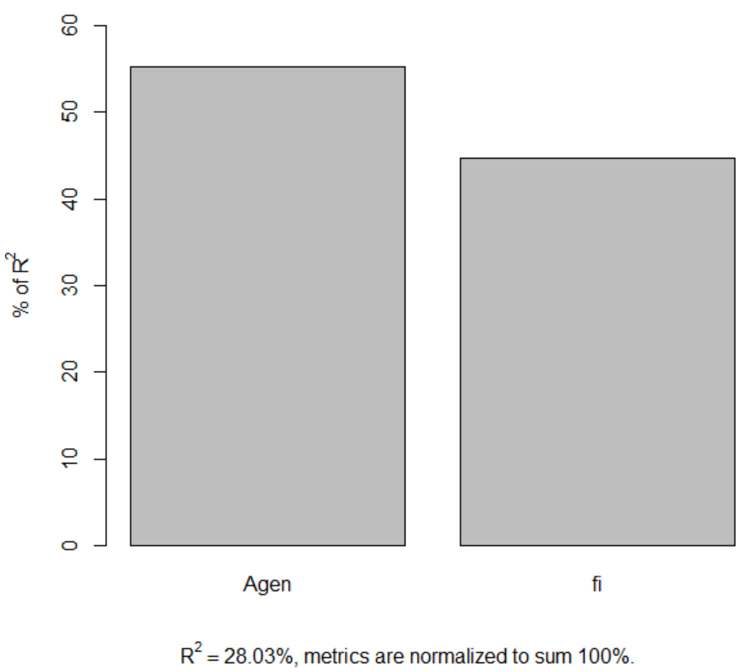


FIGURE C.8: Relative importance of agents and objective functions for renewable resources

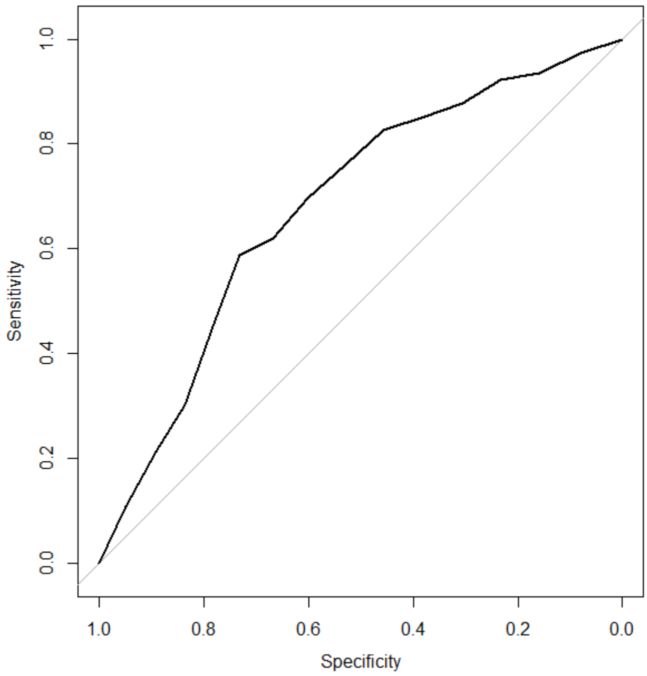


FIGURE C.9: Sensitivity and specificity diagram of the logit model (Area under the curve: 0.6826)

References

1. Abdolshah, Mohammad (2014). "A review of resource-constrained project scheduling problems (RCPSP) approaches and solutions". In: *International Transaction Journal of Engineering, Management, & Applied Sciences & Technologies* 5.4, pp. 253–286.

2. Adelakun, Olayele, Garcia, Robert, Tabaka, Ted, and Ismail, Redar (2017). "Hybrid project management: Agile with discipline". In:

3. Agarwal, Nitin and Rathod, Urvashi (2006). "Defining 'success' for software projects: An exploratory revelation". In: *International journal of project management* 24.4, pp. 358–370.

4. Aiello, Bob and Sachs, Leslie (2016). *Agile application lifecycle management: Using DevOps to drive process improvement*. Addison-Wesley Professional.

5. Ajmal, Mian M, Khan, Mehmood, Gunasekaran, Angappa, and Helo, Petri T (2022). "Managing project scope creep in construction industry". In: *Engineering, Construction and Architectural Management* 29.7, pp. 2786–2809.

6. Akgun, Zuleyha, Yilmaz, Murat, and Clarke, Paul (2020). "Assessing application lifecycle management (ALM) potentials from an industrial perspective". In: *Systems, Software and Services Process Improvement: 27th European Conference, EuroSPI 2020, Düsseldorf, Germany, September 9–11, 2020, Proceedings 27*. Springer, pp. 326–338.

7. Akinsola, Jide ET, Ogunbanwo, Afolakemi S, Okesola, Olatunji J, Odun-Ayo, Isaac J, Ayegbusi, Florence D, and Adebiyi, Ayodele A (2020). "Comparative analysis of software development life cycle models (SDLC)". In: *Intelligent Algorithms in Software Engineering: Proceedings of the 9th Computer Science On-line Conference 2020, Volume 1 9*. Springer, pp. 310–322.

8. Alba, Enrique and Chicano, J Francisco (2007). "Software project management with GAs". In: *Information sciences* 177.11, pp. 2380–2401.

9. Albtoush, AM Faten and Doh, SI (2019). "A Review on causes of cost overrun in the construction projects". In: *Int. J. New Innov. Eng. Technol* 12.3, pp. 015–022.

10. Alfieri, Arianna, Tolio, Tullio, and Urgo, Marcello (Sept. 2012). "A two-stage stochastic programming project scheduling approach to production planning". In: *The International Journal of Advanced Manufacturing Technology* 62.1, pp. 279–290. ISSN: 1433-3015. DOI: 10.1007/s00170-011-3794-4. URL: https://doi.org/10.1007/s00170-011-3794-4.

11. Algarni, Mohammed Ayedh (2021). "Contract Lifecycle Management: Processes and Benefits". In: *Innovative and Agile Contracting for Digital Transformation and Industry 4.0*. IGI Global, pp. 62–85.

12. Andreessen, Marc (2011). "Why software is eating the world". In: *Wall Street Journal* 20.2011, p. C2.

13. Arefazar, Yasaman, Nazari, Ahad, Hafezi, Mohammad Reza, and Maghool, Sayyed Amir Hossain (2022). "Prioritizing agile project management strategies as a change management tool in construction projects". In: *International Journal of Construction Management* 22.4, pp. 678–689.

14. Arya, Anjali et al. (2011a). *Application Management: Challenges-Service Creation-Strategies*. Springer.

15. Arya, Anjali et al. (2011b). *Application Management: Challenges-Service Creation-Strategies*. Springer.

16. Aytekin, Ahsen Ikbal, Tüzün, Eray, Macit, Yagup, and Tekinerdogan, Bedir (2015). "Uygulama Yaşam Döngüsü Yönetimi Sistematik Eşleme Çalışması." In: *UYMS*.

17. Babu, A.J.G. and Suresh, Nalina (1996). "Project management with time, cost, and quality considerations". In: *European Journal of Operational Research* 88.2, pp. 320–327. ISSN: 0377-2217. DOI: `https://doi.org/10.1016/0377-2217(94)00202-9`. URL: `http://www.sciencedirect.com/science/article/pii/0377221794002029`.

18. Bahadorestani, Amir, Karlsen, Jan Terje, and Motahari Farimani, Nasser (2019). "A comprehensive stakeholder-typology model based on salience attributes in construction projects". In: *Journal of Construction Engineering and Management* 145.9, p. 04019048.

19. Banjanin, Goran and Strahonja, Vjeran (2018). "Factors of Risk Reduction in Agile and Lean Enabled Governance of IT Project Portfolios". In:

20. Barafort, Béatrix, Mesquida, Antoni-Lluís, and Mas, Antónia (2019). "ISO 31000-based integrated risk management process assessment model for IT organizations". In: *Journal of Software: Evolution and Process* 31.1, e1984.

21. Barata, José and Camarinha-Matos, Luis M (2002). "Contract management in agile manufacturing systems". In: *Collaborative Business Ecosystems and Virtual Enterprises: IFIP TC5/WG5. 5 Third Working Conference on Infrastructures for Virtual Enterprises (PRO-VE'02) May 1–3, 2002, Sesimbra, Portugal 3*. Springer, pp. 109–122.

22. Batselier, Jordy and Vanhoucke, Mario (2015). "Construction and evaluation framework for a real-life project database". In: *International Journal of Project Management* 33.3, pp. 697–710.

23. Batselier and Vanhoucke (2015). "Construction and evaluation framework for a real-life project database". In: *International Journal of Project Management* 33.3, pp. 697–710. ISSN: 0263-7863. DOI: `https://doi.org/10.1016/j.ijproman.2014.09.004`. URL: `http://www.sciencedirect.com/science/article/pii/S0263786314001410`.

24. Beck, Kent et al. (2001). *Manifesto for Agile Software Development*. URL: `http://www.agilemanifesto.org/`.

25. Beek, T van der, Souravlias, D, Essen, JT van, Pruyn, J, and Aardal, K (2024). "Hybrid differential evolution algorithm for the resource constrained project scheduling problem with a flexible project structure and consumption and production of resources". In: *European Journal of Operational Research* 313.1, pp. 92–111.

26. Bergmann, Thomas and Karwowski, Waldemar (2019). "Agile Project Management and Project Success: A Literature Review". In: *Advances in Human Factors, Business Management and Society*. Ed. by Jussi Ilari Kantola, Salman Nazir, and Tibor Barath. Cham: Springer International Publishing, pp. 405–414. ISBN: 978-3-319-94709-9.

27. *BMW ConnectedDrive* (2023). URL: https://www.bmw.de/de/shop/ls/cp/connected-drive.

28. Boctor, Fayez Fouad (1993). "Heuristics for scheduling projects with resource restrictions and several resource-duration modes". In: *The international journal of production research* 31.11, pp. 2547–2558. DOI: 10.1080/00207549308956882.

29. Boell, Sebastian and Cecez-Kecmanovic, Dubravka (June 2015). "On being 'systematic' in literature reviews in IS". In: *Journal of Information Technology* 30. DOI: 10.1057/jit.2014.26.

30. Brewer, Marilynn B and Crano, William D (2000). "Research design and issues of validity". In: *Handbook of research methods in social and personality psychology*, pp. 3–16.

31. Browning, Tyson R and Yassine, Ali A (2010). "Resource-constrained multi-project scheduling: Priority rule performance revisited". In: *International Journal of Production Economics* 126.2, pp. 212–228.

32. Brucker, Peter, Drexl, Andreas, Mohring, Rolf, Neumann, Klaus, and Pesch, Erwin (Jan. 1999). "Resource-constrained project scheduling: Notation, classification, models, and methods". In: *European Journal of Operational Research* 112.1, pp. 3–41. URL: http://ideas.repec.org/a/eee/ejores/v112y1999i1p3-41.html.

33. Brusa, Eugenio, Calá, Ambra, and Ferretto, Davide (2018). "Systems Engineering and Product Lifecycle Management (PLM)". In: *Systems Engineering and Its Application to Industrial Product Development*. Springer, pp. 327–341.

34. Buganová, Katarína and Šimíčková, Jana (2019). "Risk management in traditional and agile project management". In: *Transportation Research Procedia* 40. TRANSCOM 2019 13th International Scientific Conference on Sustainable, Modern and Safe Transport, pp. 986–993. ISSN: 2352-1465. DOI: https://doi.org/10.1016/j.trpro.2019.07.138. URL: https://www.sciencedirect.com/science/article/pii/S2352146519303060.

35. Bundtzen, Henning and Hinrichs, Gerriet (2021). "The Link Between Organizational Agility And VUCA–An Agile Assessment Model". In:

36. Burgelman and Vanhoucke (2019). "Computing project makespan distributions: Markovian PERT networks revisited". In: *Computers & Operations Research* 103, pp. 123–133. ISSN: 0305-0548. DOI: https://doi.org/10.1016/j.cor.2018.10.017. URL: http://www.sciencedirect.com/science/article/pii/S0305054818302764.

37. Carmignani, Gionata, Cimino, MARIO GIOVANNI COSIMO ANTONIO, Failli, Franco, Santone, Antonella, Vaglini, Gigliola, et al. (2017). "A new perspective on Process-oriented Software Engineering based on BPMN Process Mining". In: *Atti del XIII Convegno dell'Associazione Italiana di Tecnologia Meccanica AITEM 2017*, pp. 1–10.

38. Carrillo, John and McKorkle, Scott (2008). "Application Lifecycle Management Meets Model-Driven Development-The combination of ALM and MDD gives you the connected workflow you need to handle the development of even the most complex". In: *Dr Dobb's Journal-Software Tools for the Professional Programmer*, pp. 20–25.

39. Castaneda, R Riascos, Ostrosi, E, Majić, T, Stjepandić, J, and Sagot, J-C (2020). "A method to explore product risk in product lifecycle management of configured products". In: *Proceedings of the design society: DESIGN conference*. Vol. 1. Cambridge University Press, pp. 687–696.

40. Chadli, Saad Yasser and Idri, Ali (2017). "Identifying and Mitigating Risks of Software Project Management in Global Software Development". In: *Proceedings of the 27th International Workshop on Software Measurement and 12th International Conference on Software Process and Product Measurement*. IWSM Mensura '17. Gothenburg, Sweden: Association for Computing Machinery, pp. 12–22. ISBN: 9781450348539. DOI: 10.1145/3143434.3143453. URL: https://doi.org/10.1145/3143434.3143453.

41. Chanda, Sandeep, Foggon, Damien, Chanda, Sandeep, and Foggon, Damien (2013). "Application lifecycle management". In: *Beginning ASP. NET 4.5 Databases*, pp. 235–249.

42. Chapin, Ned, Hale, Joanne E, Khan, Khaled Md, Ramil, Juan F, and Tan, Wui-Gee (2001). "Types of software evolution and software maintenance". In: *Journal of software maintenance and evolution: Research and Practice* 13.1, pp. 3–30.

43. Chappell, David et al. (2010). "What is application lifecycle management?" In: *Chappell & Associates*, p. 6.

44. Charvat, Jason (2003). "Project management methodologies: selecting, implementing, and supporting methodologies and processes for projects". In:

45. Chatterjee, Kajal, Hossain, Sheikh Ahmed, and Kar, Samarjit (June 2018). "Prioritization of project proposals in portfolio management using fuzzy AHP". In: *Operations Research* 55.2, pp. 478–501. ISSN: 0975-0320. DOI: 10.1007/s12597-018-0331-3. URL: https://doi.org/10.1007/s12597-018-0331-3.

46. Cheng, Yu Chin (2010). "Rapid Application Lifecycle Management: a new Approach with Tool Support." In:

47. Cho, Chung-Suk and Gibson Jr, G Edward (2001). "Building project scope definition using project definition rating index". In: *Journal of architectural engineering* 7.4, pp. 115–125.

48. Choetkiertikul, Morakot and Sunetnanta, Thanwadee (2012). "A risk assessment tool using a CMMI Quantitative Approach". In: *International Journal of Engineering and Technology* 4.4, p. 352.

49. Ciric, Danijela, Lalic, Bojan, Gracanin, Danijela, Palcic, Iztok, and Zivlak, Nikola (2018). "Agile Project Management in New Product Development and Innovation Processes: Challenges and Benefits Beyond Software Domain". In: *2018 IEEE International Symposium on Innovation and Entrepreneurship (TEMS-ISIE)*, pp. 1–9. DOI: 10.1109/TEMS-ISIE.2018.8478461.

50. Ciric, Danijela, Lalic, Bojan, Gracanin, Danijela, Tasic, Nemanja, Delic, Milan, and Medic, Nenad (2019). "Agile vs. Traditional approach in project management: Strategies, challenges and reasons to introduce agile". In: *Procedia Manufacturing* 39, pp. 1407–1414.

51. Clancy, Tom (1995). "The chaos report". In: *The Standish Group*.

52. Cooper, Robert G (2006). "Winning at new products: pathways to profitable innovation". In: *Proceedings Project Management Research Conference, Montreal, Canada*.

53. Corallo, Angelo, Latino, Maria Elena, Menegoli, Marta, and Pontrandolfo, Pierpaolo (2020). "A systematic literature review to explore traceability and lifecycle relationship". In: *International Journal of Production Research* 58.15, pp. 4789–4807.

54. Creemers, Stefan (July 2015). "Minimizing the expected makespan of a project with stochastic activity durations under resource constraints". In: *Journal of Scheduling* 18.3, pp. 263–273. ISSN: 1099-1425. DOI: 10.1007/s10951-015-0421-5. URL: https://doi.org/10.1007/s10951-015-0421-5.

55. Dalcher, Darren (2009). "Managing complex projects: A new model". In: *Project Management Journal* 40.3, p. 83. ISSN: 1938-9507. DOI: 10.1002/pmj.20134. URL: http://dx.doi.org/10.1002/pmj.20134.

56. Damasceno, Ernani, Azevedo, Ana, and Perez-Cota, Manuel (2022). "The State-of-the-Art of Business Intelligence and Data Mining in the Context of Grid and Utility Computing: A PRISMA Systematic Review". In: *New Perspectives in Software Engineering*. Ed. by Jezreel Mejia, Mirna Muñoz, Álvaro Rocha, Himer Avila-George, and Gloria Monica" Martínez-Aguilar. "Cham": Springer International Publishing, pp. 83–96. ISBN: 978-3-030-89909-7.

57. Dan, Schilling Nguyen (2016). "Success Factors That Influence Agile Software Development Project Success". In: *American Scientific Research Journal for Engineering, Technology, and Sciences* 17.1, pp. 172–222.

58. Danilovic, Mike and Browning, Tyson R (2007). "Managing complex product development projects with design structure matrices and domain mapping matrices". In: *International journal of project management* 25.3, pp. 300–314.

59. De Simone, Vincenzo, Amalfitano, Domenico, and Fasolino, Anna Rita (2018). "Exploiting alm and mde for supporting questionnaire-based gap analysis processes". In: *2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, pp. 1–8.

60. Dearle, Alan (2007). "Software deployment, past, present and future". In: *Future of Software Engineering (FOSE'07)*. IEEE, pp. 269–284.

61. Debels and Vanhoucke (2007). "A decomposition-based genetic algorithm for the resource-constrained project-scheduling problem". In: *Operations Research* 55.3, pp. 457–469. DOI: 10.1287/opre.1060.0358.

62. Demeulemeester (1996). "Optimal procedures for the discrete time/cost trade-off problem in project networks". In: *European Journal of Operational Research* 88.1, pp. 50–68. ISSN: 0377-2217. DOI: http://dx.doi.org/10.1016/0377-2217(94)00181-2. URL: http://www.sciencedirect.com/science/article/pii/0377221794001812.

63. Demeulemeester and Herroelen (2006). *Project scheduling: a research handbook*. Vol. 49. Springer Science & Business Media.

64. Demeulemeester, Vanhoucke, and Herroelen (Jan. 2003). "RanGen: A Random Network Generator for Activity-on-the-Node Networks". In: *Journal of Scheduling* 6.1, pp. 17–38. ISSN: 1099-1425. DOI: 10.1023/A:1022283403119. URL: https://doi.org/10.1023/A:1022283403119.

65. DeToro, Irving and McCabe, Thomas (1997). "How to stay flexible and elude fads". In: *Quality Progress* 30.3, p. 55.

66. Deuter, Andreas and Imort, Sebastian (2021). "Product lifecycle management with the asset administration shell". In: *Computers* 10.7, p. 84.

67. Deuter, Andreas, Otte, Andreas, Ebert, Marcel, and Possel-Dölken, Frank (2019). "Developing the Requirements of a PLM/ALM Integration: An Industrial Case Study". In: *Product Lifecycle Management (Volume 4): The Case Studies*, pp. 125–143.

68. Deuter, Andreas and Rizzo, Stefano (2016). "A critical view on PLM/ALM convergence in practice and research". In: *Procedia Technology* 26, pp. 405–412.

69. Diem, Gregor et al. (2021). "Agile and traditional project management: comparing agile, traditional and hybrid project management practices". PhD thesis. Heriot-Watt University.

70. Dingsøyr, Torgeir, Nerur, Sridhar, Balijepally, VenuGopal, and Moe, Nils Brede (2012). "A decade of agile methodologies: Towards explaining agile software development". In: *Journal of Systems and Software* 85.6. Special Issue: Agile Development, pp. 1213–1221. ISSN: 0164-1212. DOI: https://doi.org/10.1016/j.jss.2012.02.033. URL: http://www.sciencedirect.com/science/article/pii/S0164121212000532.

71. Dooley, Lawrence, Lupton, Gary, and O'Sullivan, David (2005). "Multiple project management: a modern competitive necessity". In: *Journal of Manufacturing Technology Management*.

72. Dove, Rick, Schindel, William (Bill), and Garlington, Ken (2018). "Case Study: Agile Systems Engineering at Lockheed Martin Aeronautics Integrated Fighter Group". In: *INCOSE International Symposium* 28.1, pp. 303–320. DOI: 10.1002/j.2334-5837.2018.00483.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/j.2334-5837.2018.00483.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/j.2334-5837.2018.00483.x.

73. Doyle, C (2007). "The importance of ALM for aerospace and defence (A&D)". In: *Embedded System Engineering (ESE magazine)* 15.5, pp. 28–29.

74. Duda, Jan, Oleszek, Sylwester, and Santarek, Krzysztof (2022). "Product lifecycle management (PLM) in the context of industry 4.0". In: *International Scientific-Technical Conference MANUFACTURING*. Springer, pp. 171–185.

75. Duka, D. (May 2012). "Agile experiences in software development". In: *2012 Proceedings of the 35th International Convention MIPRO*, pp. 692–697.

76. Dvir, Dov and Lechler, Thomas (2004). "Plans are nothing, changing plans is everything: the impact of changes on project success". In: *Research Policy* 33.1, pp. 1–15. ISSN: 0048-7333. DOI: http://dx.doi.org/10.1016/j.respol.2003.04.001. URL: http://www.sciencedirect.com/science/article/pii/S0048733303001446.

77. DŽANIĆ, Amel, Toroman, Amel, and DŽANIĆ, Alma (2022). "AGILE SOFTWARE DEVELOPMENT: MODEL, METHODS, ADVANTAGES AND DISADVANTAGES." In: *Acta Technica Corviniensis-Bulletin of Engineering* 15.4.

78. Ebert, Christof (2013). "Improving engineering efficiency with PLM/ALM". In: *Software & Systems Modeling* 12.3, pp. 443–449.

79. Ebert, Christof, Gallardo, Gorka, Hernantes, Josune, and Serrano, Nicolas (2016). "DevOps". In: *Ieee Software* 33.3, pp. 94–100.

80. Elsawah, Hany, Bakry, Ibrahim, and Moselhi, Osama (2016). "Decision Support Model for Integrated Risk Assessment and Prioritization of Intervention Plans of Municipal Infrastructure". In: *Journal of Pipeline Systems Engineering and Practice* 7.4, p. 04016010. DOI: 10.1061/(ASCE)PS.1949-1204.0000245. URL: https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29PS.1949-1204.0000245.

81. Eppinger, Steven D and Browning, Tyson R (2012). *Design structure matrix methods and applications*. MIT press.

82. Eynde, Van and Vanhoucke (2020). "Resource-constrained multi-project scheduling: benchmark datasets and decoupled scheduling". In: *Journal of Scheduling* 23.3, pp. 301–325.

83. Fang and Marle (2012). "A simulation-based risk network model for decision support in project risk management". In: *Decision Support Systems* 52.3, pp. 635–644. ISSN: 0167-9236. DOI: http://dx.doi.org/10.1016/j.dss.2011.10.021. URL: http://www.sciencedirect.com/science/article/pii/S0167923611002016.

84. Fang, Yanfeng, Wang, Weida, Yang, Chao, Zhang, Yuhang, and Chen, Zhentao (2023). "Brake-by-wire architecture design and analysis in accordance with functional safety standard". In: *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, p. 09544070231185192.

85. Fernandez, D. and Fernandez, J. (2008). "Agile project management–agilism versus traditional approaches". In: *Journal of Computer Information Systems* 49.2, pp. 10–17.

86. Fitzgerald, Brian and Stol, Klaas-Jan (2017). "Continuous software engineering: A roadmap and agenda". In: *Journal of Systems and Software* 123, pp. 176–189.

87. Forward, Andrew and Lethbridge, Timothy C (2008). "A taxonomy of software types to facilitate search and evidence-based software engineering". In: *Proceedings of the 2008 conference of the center for advanced studies on collaborative research: meeting of minds*, pp. 179–191.

88. Fowler, Martin, Highsmith, Jim, et al. (2001). "The agile manifesto". In: *Software development* 9.8, pp. 28–35.

89. Fu, Yun, Li, Minqiang, and Chen, Fuzan (2012). "Impact propagation and risk assessment of requirement changes for software development projects based on design structure matrix". In: *International Journal of Project Management* 30.3, pp. 363–373. ISSN: 0263-7863. DOI: `https://doi.org/10.1016/j.ijproman.2011.08.004`. URL: `http://www.sciencedirect.com/science/article/pii/S0263786311001062`.

90. Gaborov, Maja, Karuović, Dijana, Kavalić, Mila, Radosav, Dragica, Milosavljev, Dragana, Stanisaljev, Sanja, and Bushati, Jozef (2021). "Comparative analysis of agile and traditional methodologies in IT project management". In: *Journal of Applied Technical and Educational Sciences* 11.4, pp. 1–24.

91. Garetti, Marco and Terzi, Sergio (2003). "Product lifecycle management definizione, caratteristiche e questioni aperte". In: *Tech. Rep., Politecnico di Milano*. Politecnico de Milano, p. 154.

92. Garousi, Vahid, Tarhan, Ayça, Pfahl, Dietmar, Coşkunçay, Ahmet, and Demirörs, Onur (2019). "Correlation of critical success factors with success of software projects: an empirical investigation". In: *Software Quality Journal* 27, pp. 429–493.

93. Gasik, Stanisław (2015). "An analysis of knowledge management in PMBOK guide". In: *PM World Journal* 4.1, pp. 1–13.

94. Gatrell, Matt (2016). "The value of a single solution for end-to-end alm tool support". In: *IEEE Software* 33.5, pp. 103–105.

95. Gemino, Andrew, Horner Reich, Blaize, and Serrador, Pedro M (2021). "Agile, traditional, and hybrid approaches to project success: is hybrid a poor second choice?" In: *Project Management Journal* 52.2, pp. 161–175.

96. George, Crispin (2020). "The essence of risk identification in project risk management: An overview". In: *International Journal of Science and Research (IJSR)* 9.2, pp. 1553–1557.

97. George, Scheibe, Townsend, and Mennecke (2018). "The amorphous nature of agile: no one size fits all". In: *Journal of Systems and Information Technology* 20.2, pp. 241–260. DOI: `10.1108/JSIT-11-2017-0118`. eprint: `https://doi.org/10.1108/JSIT-11-2017-0118`. URL: `https://doi.org/10.1108/JSIT-11-2017-0118`.

98. Gerwin, Donald (1987). "An agenda for research on the flexibility of manufacturing processes". In: *International Journal of Operations & Production Management* 7.1, pp. 38–49.

99. Görög, Mihály (2002). "Strategy-Oriented Approach to Projects and the Question of Project Success". In: *Society and Economy* 24.1, pp. 55–68.

100. Goth, Greg (2009). "Agile tool market growing with the philosophy". In: *IEEE software* 26.2, pp. 88–91.

101. Göthe, Mats (2008). *Collaborative application lifecycle management with IBM rational products*. IBM Redbooks.

102. Government Commerce, OGC-Office of (2007). *The official introduction to the ITIL service lifecycle*. The Stationery Office.

103. Grant, Maria J and Booth, Andrew (2009). "A typology of reviews: an analysis of 14 review types and associated methodologies". In: *Health Information & Libraries Journal* 26.2, pp. 91–108.

104. Grey, Johannes (2011). "The development of a hybrid agile project management methodology". PhD thesis. North-West University.

105. Guendert, Steve (2011). *Mainframe history and the first users' groups(share)*.

106. Gumiński, Adam, Dohn, Katarzyna, and Oloyede, Enioluwa (2023). "Advantages and disadvantages of traditional and agile methods in software development projects–case study". In: *Zeszyty Naukowe. Organizacja i Zarządzanie/Politechnika Śląska* 188 Nowoczesność przemysłu i usług= Modernity of industry and services, pp. 191–206.

107. Gusenbauer, Michael (2019). "Google Scholar to overshadow them all? Comparing the sizes of 12 academic search engines and bibliographic databases". In: *Scientometrics* 118.1, pp. 177–214.

108. Gustavsson, Tomas (2016). "Benefits of agile project management in a non-software development context: A literature review". In: *Fifth International Scientific Conference on Project Management in the Baltic Countries, April 14-15, 2016, Riga, University of Latvia*. Latvijas Universitate, pp. 114–124.

109. Haar, Ralph-Heiner (2021). *Decoupling Hardware and Software Accelerates Development*. Tech. rep. APTIVE. URL: `https://www.aptiv.com/docs/default-source/white-papers/2021_aptiv_whitepaper_decouplinghardware.pdf?sfvrsn=623f1838_1`.

110. Habibi, Farhad, Barzinpour, Farnaz, and Sadjadi, Seyed (2018). "Resource-constrained project scheduling problem: review of past and recent developments". In: *Journal of project management* 3.2, pp. 55–88.

111. Halevi, Gali, Moed, Henk, and Bar-Ilan, Judit (2017). "Suitability of Google Scholar as a source of scientific information and as a source of data for scientific evaluation—Review of the Literature". In: *Journal of Informetrics* 11.3, pp. 823–834. ISSN: 1751-1577. DOI: `https://doi.org/10.1016/j.joi.2017.06.005`. URL: `https://www.sciencedirect.com/science/article/pii/S1751157717300676`.

112. Hallerstede, Stefan H (2013). "Open innovation platforms". In: *Managing the lifecycle of open Innovation platforms*. Springer Gabler, Wiesbaden, pp. 18–34.

113. Hansen, Anne-Sofie, Svejvig, Per, and Hansen, Lars K (2022). "Revisiting Shenhar and Dvir's Diamond Model: Do We Need an Upgrade?" In: *Research on Project, Programme and Portfolio Management: Projects as an Arena for Self-Organizing*. Springer, pp. 175–190.

114. Hartmann, Sönke and Briskorn, Dirk (2021). "An Updated Survey of Variants and Extensions of the Resource-Constrained Project Scheduling Problem". In: *European Journal of Operational Research*.

115. Harzing and Alakangas (2015). "Google Scholar, Scopus and the Web of Science: a longitudinal and cross-disciplinary comparison". In: *Scientometrics*.

116. Hauder, Viktoria A., Beham, Andreas, Raggl, Sebastian, Parragh, Sophie N., and Affenzeller, Michael (2020). "Resource-constrained multi-project scheduling with activity and time flexibility". In: *Computers & Industrial Engineering* 150, p. 106857. ISSN: 0360-8352. DOI: `https://doi.org/10.1016/j.cie.2020.106857`.

117. Heindl, Matthias, Reinisch, Franz, and Biffl, Stefan (2007). "Requirements management infrastructures in global software development-towards application lifecycle management with role-based in-time notification". In: *International Conference on Global Software Engineering (ICGSE), Workshop on Tool-Supported Requirements Management in Distributed Projects (REMIDI)*. REMIDI, ICGSE.

118. Hidalgo, Enric Senabre (2019). "Adapting the scrum framework for agile project management in science: case study of a distributed research initiative". In: *Heliyon* 5.3, e01447.

119. Hofacker, Charles F (2019). "The Growing Importance of Software as a Driver of Value Exchange". In: *Marketing in a Digital World*. Emerald Publishing Limited, pp. 85–95.

120. Homberger, Jörg (2007). "A multi-agent system for the decentralized resource-constrained multi-project scheduling problem". In: *International Transactions in Operational Research* 14.6, pp. 565–589.

121. Hu, Yong, Zhang, Xiangzhou, Ngai, E.W.T., Cai, Ruichu, and Liu, Mei (2013). "Software project risk analysis using Bayesian networks with causality constraints". In: *Decision Support Systems* 56, pp. 439–449. ISSN: 0167-9236. DOI: `http://dx.doi.org/10.1016/j.dss.2012.11.001`. URL: `http://www.sciencedirect.com/science/article/pii/S0167923612003338`.

122. Huchzermeier, Arnd and Loch, Christoph H (2001). "Project management under risk: Using the real options approach to evaluate flexibility in R… D". In: *Management Science* 47.1, pp. 85–101.

123. Hughes, D. Laurie, Dwivedi, Yogesh K., Simintiras, Antonis C., and Rana, Nripendra P. (2016). *Success and Failure of IS/IT Projects: A State of the Art Analysis and Future Directions*. 1st ed. SpringerBriefs in Information Systems. Springer International Publishing. ISBN: 978-3-319-22999-7,978-3-319-23000-9. DOI: `10.1007/978-3-319-23000-9`. URL: `https://link.springer.com/book/10.1007%2F978-3-319-23000-9`.

124. Hummer, Waldemar, Muthusamy, Vinod, Rausch, Thomas, Dube, Parijat, El Maghraoui, Kaoutar, Murthi, Anupama, and Oum, Punleuk (2019). "Modelops: Cloud-based lifecycle management for reliable and trusted ai". In: *2019 IEEE International Conference on Cloud Engineering (IC2E)*. IEEE, pp. 113–120.

125. Ilin, Igor, Levina, Anastasia, Frolov, Konstantin, Borremans, Alexandra, Ershova, Alyona, Tick, Andrea, and Averina, Mariia (2022). "Life-Cycle Contract as an Innovative Business Model for High-Tech Medical Organizations". In: *Journal of Open Innovation: Technology, Market, and Complexity* 8.4, p. 207.

126. Inflectra (2024). *List Of The Best ALM Tools*. Accessed: 2024-06-29. URL: `https://www.inflectra.com/tools/test-management/top-12-alm-tools` (visited on 03/19/2024).

127. Ishwaran, Hemant, Kogalur, Udaya B., Chen, Xi, and Minn, Andy J. (2011). "Random survival forests for high-dimensional data". In: *Statistical Analysis and Data Mining: The ASA Data Science Journal* 4.1, pp. 115–132. DOI: `10.1002/sam.10103`. eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1002/sam.10103`. URL: `https://onlinelibrary.wiley.com/doi/abs/10.1002/sam.10103`.

128. ISO (2002). *ISO/IEC 12207:1995/Amd 2:2002 Information technology – Software life cycle processes*. Tech. rep. International Organization for Standardization.

129. ISO (2018). *ISO 31000:2018 Risk management - Guidelines*. Tech. rep.

130. Issa, S and Tu, Yiliu (2020). "A survey in the resource-constrained project and multi-project scheduling problems". In: *Journal of Project Management* 5.2, pp. 117–138.

131. Jackson, Simon (2002). "Project cost overruns and risk management". In: *Proceedings of Association of Researchers in Construction Management 18th Annual ARCOM Conference, Newcastle, Northumber University, UK*. Vol. 1, pp. 1–10.

132. Jakab, Róbert and Novák, Gergely Lajos (2018). "PROJECT MANAGEMENT APPROACHES IN APPLICATION MANAGEMENT SERVICES". In: *Chapters from the Academic Aspect of Project Management-Research and Teaching Methodologies Volume II*, p. 152.

133. Jamous, Naoum, Bosse, Sascha, Görling, Carsten, Hintsch, Johannes, Khan, Ateeq, Kramer, Frederik, Müller, Hendrik, and Turowski, Klaus (2016). "Towards an IT service lifecycle management (ITSLM) Concept". In: *2016 4th International Conference on Enterprise Systems (ES)*. IEEE, pp. 29–38.

134. Kääriäinen, Jukka (2011). "Towards an application lifecycle management framework". In:

135. Kääriäinen, Jukka, Eskeli, Juho, Teppola, Susanna, Välimäki, Antti, Tuuttila, Pekka, and Piippola, Markus (2009). "Extending global tool integration environment towards lifecycle management". In: *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*. Springer, pp. 238–247.

136. Kääriäinen, Jukka and Välimäki, Antti (2008). "Impact of Application Lifecycle Management—A Case Study". In: *Enterprise Interoperability III*. Springer, pp. 55–67.

137. Kaiser, S. (2005). *Application Management in Deutschland*. Accessed on: 24.08.2010.

138. Kellenbrink, Carolin and Helber, Stefan (2015). "Scheduling resource-constrained projects with a flexible project structure". In: *European Journal of Operational Research* 246.2, pp. 379–391.

139. Kerzner, Harold (2017). *Project management: a systems approach to planning, scheduling, and controlling*. John Wiley & Sons.

140. Kircaburun, Kagan, Harris, Andrew, Calado, Filipa, and Griffiths, Mark D (2021). "The psychology of mukbang watching: A scoping review of the academic and non-academic literature". In: *International Journal of Mental Health and Addiction* 19.4, pp. 1190–1213.

141. Klespitz, József, Bíró, Miklós, and Kovács, Levente (2016). "Evaluation criteria for application life cycle management systems in practice". In: *2016 IEEE 11th International Symposium on Applied Computational Intelligence and Informatics (SACI)*. IEEE, pp. 469–472.

142. Kloppenborg, Timothy J, Tesch, Debbie, and Manolis, Chris (2014). "Project success and executive sponsor behaviors: Empirical life cycle stage investigations". In: *Project Management Journal* 45.1, pp. 9–20.

143. Knaster, Richard (2023). *SAFe 6.0 framework*. URL: https://scaledagileframework.com/.

144. Kolisch, Rainer and Sprecher, Arno (1997a). "{PSPLIB} - A project scheduling problem library: {OR} Software - {ORSEP} Operations Research Software Exchange Program". In: *European Journal of Operational Research* 96.1, pp. 205–216. ISSN: 0377-2217. DOI: http://dx.doi.org/10.1016/S0377-2217(96)00170-1. URL: http://www.sciencedirect.com/science/article/pii/S0377221796001701.

145. Kolisch, Rainer and Sprecher, Arno (1997b). "PSPLIB-a project scheduling problem library: OR software-ORSEP operations research software exchange program". In: *European journal of operational research* 96.1, pp. 205–216.

146. Kolisch, Rainer, Sprecher, Arno, and Drexl, Andreas (1995). "Characterization and generation of a general class of resource-constrained project scheduling problems". In: *Management Science* 41.10, pp. 1693–1703. DOI: 10.1287/mnsc.41.10.1693.

147. Komal, Bakhtawar, Janjua, Uzair Iqbal, Anwar, Fozia, Madni, Tahir Mustafa, Cheema, Muhammad Faisal, Malik, Muhammad Noman, and Shahid, Ahmad Raza (2020). "The impact of scope creep on project success: An empirical investigation". In: *IEEE Access* 8, pp. 125755–125775.

148. Kossmann, Mario (2016). *Requirements management: How to ensure you achieve what you need from your projects*. Routledge.

149. Kosztyán (2022a). "MFPP: Matrix-based flexible project planning". In: *SoftwareX* 17, p. 100973.

150. Kosztyán (2022b). "MFPP: Matrix-based flexible project planning". In: *SoftwareX* 17, p. 100973. ISSN: 2352-7110. DOI: https://doi.org/10.1016/j.softx.2022.100973. URL: https://www.sciencedirect.com/science/article/pii/S2352711022000012.

151. Kosztyan, Zsolt and Szalkai, Istvan (2020). "Multimode resource-constrained project scheduling in flexible projects". In: *Journal of Global Optimization* 76.1, pp. 211–241.

152. Kosztyán and Kiss (2011). "Matrix-based project planning methods". In: *Problems of Management in the 21st Century* 1, p. 67.

153. Kosztyán and Szalkai (2018). "Hybrid time-quality-cost trade-off problems". In: *Operations Research Perspectives* 5, pp. 306–318. ISSN: 2214-7160. DOI: https://doi.org/10.1016/j.orp.2018.09.003. URL: http://www.sciencedirect.com/science/article/pii/S2214716018300812.

154. Kosztyán and Szalkai (2020). "Multimode resource-constrained project scheduling in flexible projects". In: *Journal of Global Optimization* 76.1, pp. 211–241.

155. Kosztyán, Zsolt, Csizmadia, Tibor, and Katona, Attila (2021). "SIMILAR–Systematic iterative multilayer literature review method". In: *Journal of Informetrics* 15.1, p. 101111.

156. Kosztyán, Zsolt, Kiss, Judit, et al. (2010). "PEM–a New Matrix Method for Supporting the Logic Planning of Software Development Projects". In: *DSM 2010: Proceedings of the 12th International DSM Conference, Cambridge, UK, 22.-23.07. 2010*, pp. 97–110.

157. Kosztyán, Zsolt, Pribojszki-Németh, Anikó, and Szalkai, István (2019). "Hybrid multimode resource-constrained maintenance project scheduling problem". In: *Operations Research Perspectives* 6, p. 100129.

158. Kosztyán, Zsolt and Szalkai, Istvan (2018). "Hybrid time-quality-cost trade-off problems". In: *Operations Research Perspectives* 5, pp. 306–318.

159. Kosztyán, Zsolt T. (2020). "An Exact Algorithm for the Flexible Multilevel Project Scheduling Problem". In: *Expert Systems with Applications*, p. 113485.

160. Kosztyán, Zsolt Tibor (2015). "Exact algorithm for matrix-based project planning problems". In: *Expert Systems with Applications* 42.9, pp. 4460–4473.

161. Kreiner, Kristian (1995). "In search of relevance: Project management in drifting environments". In: *Scandinavian Journal of Management* 11.4. Project Management and Temporary Organozations, pp. 335–346. ISSN: 0956-5221. DOI: https://doi.org/10.1016/0956-5221(95)00029-U. URL: https://www.sciencedirect.com/science/article/pii/095652219500029U.

162. Kreter, Stefan, Schutt, Andreas, Stuckey, Peter J, and Zimmermann, Jürgen (2018). "Mixed-integer linear programming and constraint programming formulations for solving resource availability cost problems". In: *European Journal of Operational Research* 266.2, pp. 472–486.

163. Krueger, Charles W (2015). "Mechanical product lifecycle management meets product line engineering". In: *Proceedings of the 19th International Conference on Software Product Line*, pp. 316–320.

164. Kuchta, Dorota and Skowron, Dorota (2016). "Classification of R&D projects and selection of R&D project management concept". In: *R&D Management* 46.5, pp. 831–841.

165. Kute, Seema Suresh and Thorat, Surabhi Deependra (2014). "A review on various software development life cycle (SDLC) models". In: *International Journal of Research in Computer and Communication Technology* 3.7, pp. 778–779.

166. Kwak, Young Hoon and Ingall, Lisa (2007). "Exploring Monte Carlo simulation applications for project management". In: *Risk Management* 9.1, pp. 44–57.

167. Lacheiner, Hermann and Ramler, Rudolf (2011). "Application lifecycle management as infrastructure for software process improvement and evolution: Experience and insights from industry". In: *2011 37th EUROMICRO conference on software engineering and advanced applications*. IEEE, pp. 286–293.

168. Laufer, Alexander and Tucker, Richard L (1987). "Is construction project planning really doing its job? A critical examination of focus, role and process". In: *Construction management and economics* 5.3, pp. 243–266.

169. Layton, M.C., Ostermiller, S.J., and Kynaston, D.J. (2020). *Agile Project Management For Dummies*. Wiley. ISBN: 9781119677062. URL: `https://books.google.de/books?id=%5C_Pn6DwAAQBAJ`.

170. Leong, Jason, May Yee, Kiu, Baitsegi, Onalethata, Palanisamy, Lingesvaran, and Ramasamy, R Kanesaraj (2023). "Hybrid project management between traditional software development lifecycle and agile based product development for future sustainability". In: *Sustainability* 15.2, p. 1121.

171. Leybourne, Stephen A. (2009). "Improvisation and agile project management: a comparative consideration". In: *International Journal of Managing Projects in Business* 2.4, pp. 519–535. DOI: `10.1108/17538370910991124`. eprint: `https://doi.org/10.1108/17538370910991124`. URL: `https://doi.org/10.1108/17538370910991124`.

172. Liu, Bohan, Zhang, He, and Zhu, Saichun (2016). "An incremental V-model process for automotive development". In: *2016 23rd Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, pp. 225–232.

173. Liu, DN, Xu, Z, and Li, FF (2019). "Distributed resource constrained multi-project scheduling problem with cooperative-game based negotiation mechanism". In: *Syst. Eng.-Theory Pract.* 39.6, pp. 1507–1516.

174. Liu, Xiang, Saat, M. Rapik, and Barkan, Christopher P.L. (2013). "Integrated risk reduction framework to improve railway hazardous materials transportation safety". In: *Journal of Hazardous Materials* 260, pp. 131–140. ISSN: 0304-3894. DOI: `http://dx.doi.org/10.1016/j.jhazmat.2013.04.052`. URL: `http://www.sciencedirect.com/science/article/pii/S030438941300318X`.

175. Lock, Dennis (2020). *Project Management (10th edition)*. Routledge.

176. Madhuri, K Lakshmi, Suma, V, and Mokashi, Uma Mohan (2018). "A triangular perception of scope creep influencing the project success". In: *International Journal of Business Information Systems* 27.1, pp. 69–85.

177. Magid, Daniel (2007). "Controlling your outsourced application development with ALM: Put process in–take chaos out". In: *Strategic Direction*.

178. Majerowicz, Walt and Shinn, Stephen A (2016). "Schedule matters: Understanding the relationship between schedule delays and costs on overruns". In: *2016 IEEE Aerospace Conference*. IEEE, pp. 1–8.

179. Mañana-Rodríguez, Jorge (2015). "A critical review of SCImago journal & country rank". In: *Research evaluation* 24.4, pp. 343–354.

180. Marchenko, Artem and Abrahamsson, Pekka (2008). "Scrum in a multiproject environment: An ethnographically-inspired case study on the adoption challenges". In: *Agile 2008 Conference*. IEEE, pp. 15–26.

181. Marggi, R. (2002). "Application Operation – Definition, Prozesse, Organisation und Erfolgsfaktoren". In:

182. Markov, Georgi and Druzhinina, Olga (2011). *Towards an industrial ALM (Application Lifecycle) Tool Integration*.

183. Martin, Buchenau (Aug. 2023). *Autozulieferer: Bosch Verkauft Software für Fahrassistenzsysteme Nun separat*. URL: `https://www.handelsblatt.com/unternehmen/industrie/autozulieferer-bosch-verkauft-software-fuer-fahrassistenzsysteme-nun-separat/29352592.html`.

184. Mathworks, MATLAB (2023). "MATLAB 2023a". In: *The MathWorks: Natick, MA, USA*.

185. McNaughton, Blake, Ray, Pradeep, and Lewis, Lundy (2010). "Designing an evaluation framework for IT service management". In: *Information & management* 47.4, pp. 219–225.

186. Medina-Dominguez, Fuensanta, Sanchez-Segura, Maria-Isabel, Amescua, Antonio, and Garcia, Javier (2007). "Extending Microsoft team foundation server architecture to support collaborative product patterns". In: *Software Process Dynamics and Agility: International Conference on Software Process, ICSP 2007, Minneapolis, MN, USA, May 19-20, 2007. Proceedings*. Springer, pp. 1–11.

187. Meredith, J. and Mantel, S. (2019). *Project Management: A Managerial Approach (10th ed).* Weily.

188. Messnarz, Richard, Sechser, Bernhard, Höhn, Holger, Lehmann, Volker, and Ekert, Damjan (2018). "Assessing agile in automotive embedded development projects using automotive SPICE 3.1". In: *Systems, Software and Services Process Improvement: 25th European Conference, EuroSPI 2018, Bilbao, Spain, September 5-7, 2018, Proceedings 25*. Springer, pp. 443–455.

189. Metzger, Jonathan, Carlsson Kanyama, Annika, Wikman-Svahn, Per, Mossberg Sonnek, Karin, Carstens, Christoffer, Wester, Misse, and Wedebrand, Christoffer (2021). "The flexibility gamble: challenges for mainstreaming flexible approaches to climate change adaptation". In: *Journal of Environmental Policy & Planning* 23.4, pp. 543–558.

190. Minogue, Paschal et al. (2011). "" Gantt-Like" DSMs". In:

191. Mishra (2020). "Implication of theory of constraints in project management". In: *International Journal of Advanced Trends in Engineering and Technology* 5.1, pp. 1–13.

192. Mishra (2023). “Structured software development versus agile software development: a comparative analysis”. In: *International Journal of System Assurance Engineering and Management*, pp. 1–19.

193. Monghasemi, Shahryar, Nikoo, Mohammad Reza, Fasaee, Mohammad Ali Khaksar, and Adamowski, Jan (2015). “A novel multi criteria decision making model for optimizing time-cost-quality trade-off problems in construction projects”. In: *Expert Systems with Applications* 42.6, pp. 3089–3104. ISSN: 0957-4174. DOI: `http://dx.doi.org/10.1016/j.eswa.2014.11.032`. URL: `http://www.sciencedirect.com/science/article/pii/S0957417414007180`.

194. Moore, Richard, Reff, Kelly, Graham, James, and Hackerson, Brian (2007). “Scrum at a fortune 500 manufacturing company”. In: *Agile 2007 (AGILE 2007)*. IEEE, pp. 175–180.

195. Morales, Juan M et al. (2014). “Managing uncertainty with flexibility”. In: *Integrating Renewables in Electricity Markets: Operational Problems*, pp. 137–171.

196. Morcov, Stefanut, Pintelon, Liliane, and Kusters, Rob J (2020). “Definitions, characteristics and measures of IT project complexity-a systematic literature review”. In: *International journal of information systems and project management* 8.2, pp. 5–21.

197. Moreira, Mario E (2013). “Considering Agile Tools within an ALM Framework”. In: *Being Agile*. Springer, pp. 201–208.

198. Moukrim, Aziz, Quilliot, Alain, and Toussaint, Hélène (2015). “An effective branch-and-price algorithm for the preemptive resource constrained project scheduling problem based on minimal interval order enumeration”. In: *European Journal of Operational Research* 244.2, pp. 360–368.

199. Ng, Jun Jie and Navaretnam, Sathesh So (2019). “Contract management: being agile”. In: *IEEE Engineering Management Review* 47.3, pp. 33–35.

200. Niemann, Jörg and Pisla, Adrian (2018). “Sustainable potentials and risks assess in automation and robotization using the life cycle management index tool—LY-MIT”. In: *Sustainability* 10.12, p. 4638.

201. Odzaly, Edzreena Edza, Greer, Des, and Stewart, Darryl (June 2018). “Agile risk management using software agents”. In: *Journal of Ambient Intelligence and Humanized Computing* 9.3, pp. 823–841. ISSN: 1868-5145. DOI: `10.1007/s12652-017-0488-2`. URL: `https://doi.org/10.1007/s12652-017-0488-2`.

202. Oecking, Christian and Degenhardt, Andreas (2011). “Application Management 2.0”. In: *Application Management: Challenges–Service Creation–Strategies*. Springer, pp. 3–29.

203. Oh, Jeongsu, Yang, Jeongsam, and Lee, Sungjoo (2012). “Managing uncertainty to improve decision-making in NPD portfolio management with a fuzzy expert system”. In: *Expert Systems with Applications* 39.10, pp. 9868–9885. ISSN: 0957-4174. DOI: `http://dx.doi.org/10.1016/j.eswa.2012.02.164`. URL: `http://www.sciencedirect.com/science/article/pii/S0957417412004277`.

204. Okon, Raymond, Eleberi, Ebele Leticia, and Uka, Kanayo Kizito (2020). “A web based digital repository for scholarly publication”. In: *Journal of Software Engineering and Applications* 13.4, pp. 67–75.

205. Olsson, Nils (2006a). "Management of flexibility in projects". In: *International Journal of Project Management* 24.1, pp. 66–74. ISSN: 0263-7863. DOI: https://doi.org/10.1016/j.ijproman.2005.06.010. URL: https://www.sciencedirect.com/science/article/pii/S0263786305000773.

206. Olsson, Nils (2006b). "Management of flexibility in projects". In: *International Journal of Project Management* 24.1, pp. 66–74.

207. Otibine, Tobias, Mbuguah, Samuel, Kilwake, Humphrey, and Tsinale, Harriet (Nov. 2017). "Application Lifecycle Management Activities For Quality Assurance In Software Development". In: *International Journal of Trend in Research and Development* 4, pp. 2394–9333.

208. Page, Matthew J et al. (2021). "The PRISMA 2020 statement: an updated guideline for reporting systematic reviews". In: *International journal of surgery* 88, p. 105906.

209. Papadakis, Emmanouil and Tsironis, Loukas (2020). "Towards a hybrid project management framework: A systematic literature review on traditional, agile and hybrid techniques". In: *The Journal of Modern Project Management* 8.2.

210. Paré, Guy, Trudel, Marie-Claude, Jaana, Mirou, and Kitsiou, Spyros (2015a). "Synthesizing information systems knowledge: A typology of literature reviews". In: *Information & Management* 52.2, pp. 183–199.

211. Paré, Guy, Trudel, Marie-Claude, Jaana, Mirou, and Kitsiou, Spyros (2015b). "Synthesizing information systems knowledge: A typology of literature reviews". In: *Information & Management* 52.2, pp. 183–199.

212. Park, Jung Eun (2021). "Schedule delays of major projects: what should we do about it?" In: *Transport Reviews* 41.6, pp. 814–832.

213. Patterson, James H (1976). "Project scheduling: The effects of problem structure on heuristic performance". In: *Naval Research Logistics Quarterly* 23.1, pp. 95–123. DOI: 10.1002/nav.3800230110.

214. Pellerin, Robert and Perrier, Nathalie (2019). "A review of methods, techniques and tools for project planning and control". In: *International Journal of Production Research* 57.7, pp. 2160–2178.

215. Pereira, Rúben and Serrano, João (2020). "A review of methods used on IT maturity models development: A systematic literature review and a critical analysis". In: *Journal of Information Technology*, p. 0268396219886874.

216. Peteghem and Vanhoucke (2014). "An experimental investigation of metaheuristics for the multi-mode resource-constrained project scheduling problem on new dataset instances". In: *European Journal of Operational Research* 235.1, pp. 62–72. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2013.10.012. URL: http://www.sciencedirect.com/science/article/pii/S0377221713008357.

217. Pham, Mai T, Rajić, Andrijana, Greig, Judy D, Sargeant, Jan M, Papadopoulos, Andrew, and McEwen, Scott A (2014). "A scoping review of scoping reviews: advancing the approach and enhancing the consistency". In: *Research synthesis methods* 5.4, pp. 371–385.

218. Pich, Michael T., Loch, Christoph H., and Meyer, Arnoud De (2002). "On Uncertainty, Ambiguity, and Complexity in Project Management". In: *Management Science* 48.8, pp. 1008–1023. DOI: 10.1287/mnsc.48.8.1008.163. eprint: https://doi.org/10.1287/mnsc.48.8.1008.163. URL: https://doi.org/10.1287/mnsc.48.8.1008.163.

219. Pinto (2010). "Achieving competitive advantage". In: *New Jercy*.

220. Pirklbauer, Guenter, Ramler, Rudolf, and Zeilinger, Rene (2009). "An integration-oriented model for application lifecycle management". In: *International Conference on Enterprise Information Systems*. Vol. 1. SCITEPRESS, pp. 399–402.

221. PMI (2021). *Project management body of knowledge (PMBOK® guide)*.

222. Polit, Denise F (2004). *Nursing research: Appraising evidence for nursing practice*. Lippincott Williams & Wilkins.

223. Pollack, Julien, Helm, Jane, and Adler, Daniel (2018). "What is the Iron Triangle, and how has it changed?" In: *International journal of managing projects in business* 11.2, pp. 527–547.

224. Ponsard, Christophe, Germeau, Fabian, Ospina, Gustavo, Bitter, Jan, Mende, Hendrik, Vossen, René, and Schmitt, Robert H. (2019). "A Two-phased Risk Management Framework Targeting SMEs Project Portfolios". In: *Proceedings of the 9th International Conference on Siauthor=Christophe Ponsard and Fabian Germeau and Gustavo Ospina and Jan Bitter and Hendrik Mende and René Vossen and Robert H. Schmitt, mulation and Modeling Methodologies, Technologies and Applications - Volume 1: SIMULTECH,* INSTICC. SciTePress, pp. 406–413. ISBN: 978-989-758-381-0. DOI: 10.5220/0008119704060413.

225. Ragunath, PK, Velmourougan, S, Davachelvan, P, Kayalvizhi, S, and Ravimohan, R (2010). "Evolving a new model (SDLC Model-2010) for software development life cycle (SDLC)". In: *International Journal of Computer Science and Network Security* 10.1, pp. 112–119.

226. Raharjo, Teguh and Purwandari, Betty (2020). "Agile Project Management Challenges and Mapping Solutions: A Systematic Literature Review". In: *Proceedings of the 3rd International Conference on Software Engineering and Information Management*. ICSIM '20. Sydney, NSW, Australia: Association for Computing Machinery, pp. 123–129. ISBN: 9781450376907. DOI: 10.1145/3378936.3378949. URL: https://doi.org/10.1145/3378936.3378949.

227. Rahman, Md Humyun Fuad, Chakrabortty, Ripon K, and Ryan, Michael J (2021). "Managing uncertainty and disruptions in resource constrained project scheduling problems: a real-time reactive approach". In: *IEEE Access* 9, pp. 45562–45586.

228. Ramler, Rudolf, Lacheiner, Hermann, and Kern, Albin (2012). "Rule-based detection of process conformance violations in application lifecycle management". In: *Systems, Software and Services Process Improvement: 19th European Conference, EuroSPI 2012, Vienna, Austria, June 25-27, 2012. Proceedings 19*. Springer, pp. 37–48.

229. Rao, Purnima and Palaniappan, Kandhasami (2020). "Continuous Engineering Through ALM-PLM Integration". In: *Product Lifecycle Management Enabling Smart X: 17th IFIP WG 5.1 International Conference, PLM 2020, Rapperswil, Switzerland, July 5–8, 2020, Revised Selected Papers 17*. Springer, pp. 798–811.

230. Regan, Gilbert, Biro, Miklos, Flood, Derek, and McCaffery, Fergal (2015). "Assessing traceability—practical experiences and lessons learned". In: *Journal of Software: Evolution and Process* 27.8, pp. 591–601.

231. Reiff, Janine and Schlegel, Dennis (2022). "Hybrid project management–a systematic literature review". In: *International journal of information systems and project management* 10.2, pp. 45–63.

232. Ren, Ran, Zhang, Jiansong, and Jiang, Yi (2021). "New automated activity-on-node calculation grading method for construction management education innovation". In: *Journal of Civil Engineering Education* 147.3, p. 04021004.

233. Resing, Gregor (2023). *The Software Defined Vehicle*. Tech. rep. IBM. URL: `https://www.ibm.com/blogs/digitale-perspektive/2023/06/the-software-defined-vehicle/`.

234. Rokade, Varsha (2008). "Information Technology and its Growing Role in Retail Sector". In: *It Enabled Practices And Emerging Management Paradigms*, p. 105.

235. Rossberg, Joachim (2008). *Pro Visual Studio Team System Application Lifecycle Management*. Apress.

236. Rossberg, Joachim (2014). *Beginning application lifecycle management*. Apress.

237. Rossberg, Joachim (2016). "An Overview of TFS". In: *Agile Project Management using Team Foundation Server 2015*, pp. 19–35.

238. Rossberg, Joachim (2019). "Introduction to Application Life Cycle Management". In: *Agile Project Management with Azure DevOps*. Springer, pp. 1–36.

239. Rossman, Bruce (2010). *Application Lifecycle Management-Activities, Methodologies, Disciplines, Tools, Benefits, ALM Tools and Products*. Emereo Pty Ltd.

240. Roy, B (1962). "Cheminement et connexité dans les graphes". In: *Applications aux problémes d'ordonnancement. METRA: Série Spéciale*.1.

241. Ruparelia, Nayan B (2010). "Software development lifecycle models". In: *ACM SIGSOFT Software Engineering Notes* 35.3, pp. 8–13.

242. Sahu, Kavita, Shree, R, and Kumar, R (2014). "Risk management perspective in SDLC". In: *International Journal of Advanced Research in Computer Science and Software Engineering* 4.3.

243. Salameh, Hanadi (2014). "What, when, why, and how? A comparison between agile project management and traditional project management methods". In: *International Journal of Business and Management Review* 2.5, pp. 52–74.

244. Sánchez, Mariam Gómez, Lalla-Ruiz, Eduardo, Gil, Alejandro Fernández, Castro, Carlos, and Voß, Stefan (2022). "Resource-Constrained Multi-Project Scheduling Problem: A Survey". In: *European Journal of Operational Research*.

245. Al-Saqqa, Samar, Sawalha, Samer, and AbdelNabi, Hiba (2020). "Agile software development: Methodologies and trends." In: *International Journal of Interactive Mobile Technologies* 14.11.

246. Schwaber, Carey et al. (2006). "The changing face of application life-cycle management". In: *Forrester Research* 18.

247. Schwalbe, Kathy (2015). *Information technology project management*. Cengage Learning.

248. Serrador and Pinto (2015). "Does Agile work? — A quantitative analysis of agile project success". In: *International Journal of Project Management* 33.5, pp. 1040–1051. ISSN: 0263-7863. DOI: https://doi.org/10.1016/j.ijproman.2015.01.006. URL: https://www.sciencedirect.com/science/article/pii/S0263786315000071.

249. Servranckx, Tom and Vanhoucke, Mario (2019). "A tabu search procedure for the resource-constrained project scheduling problem with alternative subgraphs". In: *European Journal of Operational Research* 273.3, pp. 841–860.

250. Servranckx and Vanhoucke (2019). "Strategies for project scheduling with alternative subgraphs under uncertainty: similar and dissimilar sets of schedules". In: *European Journal of Operational Research* 279.1, pp. 38–53.

251. SGI (2019). *CHAOS Manifesto*. Standish Group International.

252. Shafqat, A, Oehmen, Josef, and Welo, Torgeir (2022). "Planning unplanned design iterations using risk management and learning strategies". In: *Journal of Engineering Design* 33.2, pp. 120–143.

253. Shaw, K (2007). "Application lifecycle management for the enterprise". In: *Serena Software, White Paper, http://www.serena.com/Docs/Repository/company/Serena_ALM_2. 0_For_t. pdf (available 24.04. 2008)*.

254. Shenhar, Aaron J and Dvir, Dov (2007). *Reinventing project management: the diamond approach to successful growth and innovation*. Harvard Business Review Press.

255. Sinderen, Marten J van, Halteren, Aart Tijmen van, Wegdam, Maarten, Meeuwissen, Hendrik B, and Eertink, E Henk (2006). "Supporting context-aware mobile applications: an infrastructure approach". In: *IEEE Communications Magazine* 44.9, pp. 96–104.

256. Singh, Alina and Ahlawat, Navin (2023). "A review article:-the Growing Role of Data Science and AI in Banking and Finance". In: *Int. Res. J. Mod. Eng. Technol. Sci* 08, pp. 1047–1051.

257. Sohi, Afshin Jalali, Bosch-Rekveldt, Marian, and Hertogh, Marcel (2019). "Practitioners' perspectives on flexible project management". In: *IEEE Transactions on Engineering Management* 68.4, pp. 911–925.

258. Sommer, Svenja C., Loch, Christoph H., and Dong, Jing (2009). "Managing Complexity and Unforeseeable Uncertainty in Startup Companies: An Empirical Study". In: *Organization Science* 20.1, pp. 118–133. DOI: `10.1287/orsc.1080.0369`. eprint: `https://pubsonline.informs.org/doi/pdf/10.1287/orsc.1080.0369`. URL: `https://pubsonline.informs.org/doi/abs/10.1287/orsc.1080.0369`.

259. Sonnemann, Guido, Gemechu, Eskinder Demisse, Remmen, Arne, Frydendal, Jeppe, and Jensen, Allan Astrup (2015a). "Life cycle management: Implementing sustainability in business practice". In: *Life cycle management*, pp. 7–21.

260. Sonnemann, Guido, Gemechu, Eskinder Demisse, Remmen, Arne, Frydendal, Jeppe, and Jensen, Allan Astrup (2015b). "Life cycle management: Implementing sustainability in business practice". In: *Life cycle management*, pp. 7–21.

261. Špundak, Mario (2014). "Mixed agile/traditional project management methodology–reality or illusion?" In: *Procedia-Social and Behavioral Sciences* 119, pp. 939–948.

262. StartupStash (2022). *Top 33 Application Lifecycle Management (ALM) Tools*. Accessed: 2024-06-29. URL: `https://startupstash.com/application-lifecycle-management-alm-tools/` (visited on 12/20/2022).

263. Stefanova-Stoyanova, Varbinka and Danov, Petko (2022). "Comparative Analysis of Specialized Standards and Methods on Increasing the Effectiveness and Role of PDCA for Risk Control in Management Systems". In: *2022 10th International Scientific Conference on Computer Science (COMSCI)*. IEEE, pp. 1–4.

264. Stepanović, Mia, Jovičić, Jelena, Stupar, Goran, and Kovačević, Marko (2018). "Application lifecycle management in automotive: Adaptive AUTOSAR example". In: *2018 IEEE 8th International Conference on Consumer Electronics-Berlin (ICCE-Berlin)*. IEEE, pp. 1–4.

265. Stretton, Alan (2021). "Revisiting classifications of types of projects and programs and their application sectors1". In:

266. Sullivan, Danny (Mar. 2022). *Why google can't count results properly*. URL: `https://searchengineland.com/why-google-cant-count-results-properly-53559`.

267. Sydow, Jörg and Windeler, Arnold (2020). "Temporary organizing and permanent contexts". In: *Current Sociology* 68.4, pp. 480–498.

268. Tang, Dunbing, Zhu, Renmiao, Tang, Jicheng, Xu, Ronghua, and He, Rui (2010). "Product design knowledge management based on design structure matrix". In: *Advanced Engineering Informatics* 24.2, pp. 159–166.

269. Tao, Sha and Dong, Zhijie Sasha (2017). "Scheduling resource-constrained project problem with alternative activity chains". In: *Computers & Industrial Engineering* 114, pp. 288–296.

270. Tavares, L. Valadares, Ferreira, J. Antunes, and Coelho, J. Silva (1999). "The risk of delay of a project in terms of the morphology of its network". In: *European Journal of Operational Research* 119.2, pp. 510–537. ISSN: 0377-2217. DOI: `https://doi.org/10.1016/S0377-2217(99)00150-2`. URL: `http://www.sciencedirect.com/science/article/pii/S0377221799001502`.

271. Theocharis, Georgios, Kuhrmann, Marco, Münch, Jürgen, and Diebold, Philipp (2015). "Is Water-Scrum-Fall Reality? On the Use of Agile and Traditional Development Practices". In: *Product-Focused Software Process Improvement*. Ed. by Pekka Abrahamsson, Luis Corral, Markku Oivo, and Barbara Russo. Cham: Springer International Publishing, pp. 149–166. ISBN: 978-3-319-26844-6.

272. Tritschler, Martin, Naber, Anulark, and Kolisch, Rainer (2017). "A hybrid metaheuristic for resource-constrained project scheduling with flexible resource profiles". In: *European Journal of Operational Research* 262.1, pp. 262–273.

273. Tudenhöfner, Eduard (2011). *Integration of Performance Management into the Application Lifecycle*. diplom. de.

274. Turner, J Rodney and Müller, Ralf (2003). "On the nature of the project as a temporary organization". In: *International journal of project management* 21.1, pp. 1–8.

275. Tüzün, Eray, Tekinerdogan, Bedir, Macit, Yagup, and İnce, Kürşat (2019). "Adopting integrated application lifecycle management within a large-scale software company: An action research approach". In: *Journal of Systems and Software* 149, pp. 63–82.

276. Välimäki, Antti and Kääriäinen, Jukka (2008). "Patterns for distributed scrum—a case study". In: *Enterprise interoperability III*. Springer, pp. 85–97.

277. van den Ende, Leonore and van Marrewijk, Alfons (2014). "The ritualization of transitions in the project life cycle: A study of transition rituals in construction projects". In: *International Journal of Project Management* 32.7, pp. 1134–1145. ISSN: 0263-7863. DOI: https://doi.org/10.1016/j.ijproman.2014.02.007. URL: https://www.sciencedirect.com/science/article/pii/S0263786314000295.

278. Van der Beek, T, Van Essen, JT, Pruyn, J, and Aardal, K (2022). *Exact solution methods for the resource constrained project scheduling problem with a flexible project structure*.

279. Van Wyngaard, C Jurie, Pretorius, Jan-Harm C, and Pretorius, Leon (2012). "Theory of the triple constraint—A conceptual review". In: *2012 IEEE International Conference on Industrial Engineering and Engineering Management*. IEEE, pp. 1991–1997.

280. Vanhoucke (2012). "Measuring the efficiency of project control using fictitious and empirical project data". In: *International Journal of Project Management* 30.2, pp. 252–263. ISSN: 0263-7863. DOI: https://doi.org/10.1016/j.ijproman.2011.05.006. URL: http://www.sciencedirect.com/science/article/pii/S026378631100069X.

281. Vanhoucke, Coelho, Debels, Maenhout, and Tavares (2008). "An evaluation of the adequacy of project network generators with systematically sampled networks". In: *European Journal of Operational Research* 187.2, pp. 511–524. DOI: doi.org/10.1016/j.ejor.2007.03.032.

282. Vanhoucke, Mario, Coelho, José, Debels, Dieter, Maenhout, Broos, and Tavares, Luis V. (2008). “An evaluation of the adequacy of project network generators with systematically sampled networks”. In: *European Journal of Operational Research* 187.2, pp. 511–524. ISSN: 0377-2217. DOI: `https://doi.org/10.1016/j.ejor.2007.03.032`. URL: `http://www.sciencedirect.com/science/article/pii/S037722170700375X`.

283. Vázquez, E Pérez, Calvo, M Posada, and Ordóñez, P Martín (2015). “Learning process on priority rules to solve the RCMPSP”. In: *Journal of Intelligent Manufacturing* 26.1, pp. 123–138. DOI: `10.1007/s10845-013-0767-5`.

284. Venkatesh, Bharti and Balani, Lalit (2016). “Requirement management a key to successful project management for software systems”. In: *Voice of Research* 5.1.

285. Vresk, Anja, Pihir, Igor, and Furjan, Martina Tomičić (2020). “Agile vs. Traditional Methods for Managing IT Projects-A Case Study”. In: *Central European Conference on Information and Intelligent Systems*. Faculty of Organization and Informatics Varazdin, pp. 183–191.

286. Wagenaar, Gerard, Overbeek, Sietse, Lucassen, Garm, Brinkkemper, Sjaak, and Schneider, Kurt (July 2018). “Working software over comprehensive documentation – Rationales of agile teams for artefacts usage”. In: *Journal of Software Engineering Research and Development* 6.1, p. 7. ISSN: 2195-1721. DOI: `10.1186/s40411-018-0051-7`. URL: `https://doi.org/10.1186/s40411-018-0051-7`.

287. Wallace, Linda, Keil, Mark, and Rai, Arun (2004). “Understanding software project risk: a cluster analysis”. In: *Information & Management* 42.1, pp. 115–125. ISSN: 0378-7206. DOI: `http://dx.doi.org/10.1016/j.im.2003.12.007`. URL: `http://www.sciencedirect.com/science/article/pii/S0378720604000102`.

288. Walter, Karolak (1995). *Software Engineering Risk Management: A Just-in-Time Approach*. Los Alamitos, CA, USA: IEEE Computer Society Press. ISBN: 0818671947.

289. Wang, Tao, Gao, Shangde, Li, Xiaodong, and Ning, Xin (2018). “A Meta-network-based Risk Evaluation and Control Method for Industrialized Building Construction Projects”. In: *Journal of Cleaner Production*. ISSN: 0959-6526. DOI: `https://doi.org/10.1016/j.jclepro.2018.09.127`. URL: `http://www.sciencedirect.com/science/article/pii/S0959652618328506`.

290. Watson, Richard (2015). “Beyond being systematic in literature reviews in IS”. In: *Journal of Information Technology* 30, pp. 185–187.

291. Wawak, Sławomir, Ljevo, Žanesa, and Vukomanović, Mladen (2020). “Understanding the key quality factors in construction projects—A systematic literature review”. In: *Sustainability* 12.24, p. 10376.

292. Weiss, Gerhard et al. (2009). *Software engineering–processes and tools*. Springer.

293. Wells, Hany, Dalcher, Darren, and Smyth, Hedley (2015). “The adoption of agile management practices in a traditional project environment: An IT/IS Case Study”. In: *2015 48th Hawaii international conference on system sciences*. IEEE, pp. 4446–4453.

294. West, Dave, Gilpin, Mike, Grant, Tom, and Anderson, Alissa (2011). "Water-scrum-fall is the reality of agile for most organizations today". In: *Forrester Research* 26.

295. Westland, Jason (2007). *The project management life cycle: A complete step-by-step methodology for initiating planning executing and closing the project*. Kogan Page Publishers.

296. Whelton, Michael and Ballard, Glenn (2002). "Wicked problems in project definition". In: *Proceedings of the International Group for Lean Construction 10th Annual Conference, Brazil*.

297. Wiest, Jerome D (1981). "Precedence diagramming method: Some unusual characteristics and their implications for project managers". In: *Journal of Operations management* 1.3, pp. 121–130.

298. Wysocki, Robert K (2010). *Effective software project management*. John Wiley & Sons.

299. Wysocki, Robert K (2011a). *Effective project management: traditional, agile, extreme*. John Wiley & Sons.

300. Wysocki, Robert K (2011b). *Effective project management: traditional, agile, extreme*. John Wiley & Sons.

301. Xiao and Watson (2019). "Guidance on conducting a systematic literature review". In: *Journal of planning education and research* 39.1, pp. 93–112.

302. Yadav, Neeraj and Heriyati, Pantri (2023). "Fallacy of continual improvement in the longitudinal study of ISO 9001 and IATF 16949 standards". In: *International Journal of Productivity and Performance Management*.

303. Yazan, Bedrettin (2015). "Three approaches to case study methods in education: Yin, Merriam, and Stake". In: *The qualitative report* 20.2, pp. 134–152.

304. Yin, Robert K (2009). *Case study research: Design and methods*. Vol. 5. sage.

305. Yusof, Norhanisha et al. (2021). "The Trend of Published Literature on User Experience (UX) Evaluation: A Bibliometric Analysis". In: *Turkish Journal of Computer and Mathematics Education (TURCOMAT)* 12.3, pp. 1529–1542.

306. Zafar, Atique Ahmad, Saif, Shahela, Khan, Muzafar, Iqbal, Javed, Akhunzada, Adnan, Wadood, Abdul, Al-Mogren, Ahmad, and Alamri, Atif (2018). "Taxonomy of factors causing integration failure during global software development". In: *IEEE Access* 6, pp. 22228–22239.

307. Zhu, Jin and Mostafavi, Ali (2015). "An Integrated Framework for the Assessment of the Impacts of Uncertainty in Construction Projects Using Dynamic Network Simulation". In: *Computing in Civil Engineering 2015*. DOI: `10.1061/9780784479247.044`. eprint: `https://ascelibrary.org/doi/pdf/10.1061/9780784479247.044`. URL: `https://ascelibrary.org/doi/abs/10.1061/9780784479247.044`.

308. Zwikael, Ofer (2009). "Critical planning processes in construction projects". In: *Construction innovation*.